2019-08-02

Implementation and Design of Webcam Systems for Traffic Monitoring in Acadia National Park

Mingzhang Zhu
Worcester Polytechnic Institute

Nicholas Hackett Hollander
Worcester Polytechnic Institute

Yicheng Yang
Worcester Polytechnic Institute

Bar Harbor Project Center

Implementation and Design of Webcam Systems for Traffic Monitoring in Acadia National Park

An Interactive Qualifying Project

Submitted to the Faculty of

WORCESTER POLYTECHNIC INSTITUTE

in Partial Fulfilment of the Requirements for the

Degree of Bachelor of Science

by

Nicholas Hollander

Mingzhang Zhu

Yicheng Yang

Submitted to:

Prof. Frederick Bianchi

Date:

August 3rd, 2019

Abstract

Because traffic congestion often has impacts on safety, visitor experience, resources, emergency response times, and park operations, reducing congestion is an important management topic for Acadia National Park. This project explores the use of remote webcams to monitor parking lots and other areas of interest within the park, with the ability to upload images in real time to the Acadia Park website where they can be viewed by the public and Acadia management personnel. This strategy is being explored to help mitigate traffic congestion.

Acknowledgements

Our team would like to thank the many Acadia National Park Rangers who helped us with our project. Thank you to Jay Elhard for showing us around the North Atlantic Coastline Webcam and for providing technical details on the park web infrastructure. We would like to acknowledge Ranger John Kelly for providing us with the information of road closure and congestion statistics. We would also like to send thanks to our liaisons, Science Coordinator Abe Miller-Rushing and Conservation Director Stephanie Client, for giving valuable feedback about our implementation of the static webcam. We are also appreciative of all the support from other park staff members as well as staff from Friends of Acadia. Also thank you to the College of the Atlantic for providing housing.

At last, we want to express our sincerest thanks to our professors Frederick Bianchi and Balistrieri, Thomas J. Professor Balistrieri gave us some enlightening ideas and helped us work through the social science part of this project. Professor Bianchi was a responsible and brilliant advisor for guiding us through all the difficult and confusing times.

Executive Summary

Recent years have seen a massive expansion of visitation to Acadia National Park. In an attempt to control this influx, Acadia National Park has developed a traffic management plan, part of which requires monitoring of traffic to establish the effectiveness of their plan. To assist in the completion of this goal, we investigated a variety of autonomous webcam related systems that could be used in a number of ways to actively and passively monitor conditions within the park.

Project Goal

The goal of this project was to develop a comprehensive webcam system for monitoring the road conditions and sending back the images on websites. For park rangers, this system can help gather real-time information as an indicator of critical congestion. For visitors, they can check the current and past road conditions as a reference for planning their trip. To achieve this goal, we created a solar-powered cellular static webcam and proved that this system can continuously work in the field with minimal maintenance. We also proved the concept of a GPS activated mobile webcam system which will automatically capture images at chosen GPS coordinates and automatically upload them to a web server. In addition, a proposal design is made to demonstrate the functions of a potential web page for the use of park rangers and visitors. The primary objectives of this project were to:

- Design, develop, and construct a fully self-contained webcam system ready for deployment in the national parks.
- Implement the static webcam via installation at a location of key importance within the park.
- Design, develop, and construct a prototype mobile webcam system capable of wide-area coverage.
- Verify the concept of a mobile webcam system by conducting field tests throughout key locations in the park.
- Create a prototype web page design to display the images collected from both the static and mobile camera platforms in real time, as well as providing access to historic images.
- Evaluate the results of the previous objectives, and offer a set of recommendations for how park staff as well as future WPI projects should continue work on this subject.

Development of Methodology

In order to achieve the goals presented to us, we explored a number of methods. Development of the technical outlines for the static webcam system was undertaken with the understanding that it would need to operate under a variety of weather conditions. Many specific requirements for the static camera platform were resolved by the 2018 Acadia National Park Webcam Research Group, including the ability to operate for multiple days under adverse weather conditions.

In addition to the static camera, we investigated the unique requirements of effectively capturing data from a large number of locations which may feature little to no connectivity. An investigation of cellular connectivity conducted by the 2018 Cellular Connectivity group showed that there are many locations in the park which feature absolutely no 4G/LTE connectivity, despite containing regions of potential interest. Internal discussions resulted in the decision to implement a mobile platform for capturing the images upon the Island Explorer bus network. This allowed the camera to reach dozens of points throughout the park, including those without reception by caching the images internally for upload at a later date.

The final methodology was to create a web interface to enable members of the public and park administration to view present and historical information from the cameras. Previous commercial solutions did not permit public access to image data, and severely limited access to authorized personnel. This necessitated the creation of an accessible and intuitive interface.

Potential Impact of Methodology

This project provided the technological framework for a complete static camera based monitoring system to the National Park Service, which can be used to assist in the key monitoring aspects of Acadia National Park's transportation plan. If sufficient developments are made into a large-scale deployment of this technology, park staff will be able to actively monitor important locations in the park.

The static camera methodologies were created with the intent that park would explore options for third party contractors to perform the final implementation across the park. The development and implementation of the static camera proved that it is not only possible, but feasible to implement

and deploy an autonomous wireless camera system in the park, and gather valuable information from it.

Components of Static Webcam System

Example Image Taken by Static Webcam on Cadillac Mountain

The mobile camera methodologies were created to demonstrate the value of future research into the topic. While we were able to capture and upload images consistently from test vehicles, additional research will be required into the subject before a set of final recommendations can be offered.

System Setup on the Top of Vehicle

Example Image Taken by Mobile Webcam on Cadillac Mountain

The website methodology produced a design for an interactive website that could be used by both visitors and park staff to obtain current and past information about traffic conditions throughout a wide area of the park. If converted into a fully interactive web application, it would provide intuitive access to thousands of present and historical data points.

Based on all the images and data evaluation, we offered recommendations from three perspectives: equipment improvement, image analysis, and park implementation. We recommended that the battery capacity should be doubled for a long-term use; For specific locations a point-to-point Ethernet bridge could be a preferred choice for data transmission; A rebooting function should be

added into the static webcam system for significantly saving power. We also suggested that image recognition AI should be conducted to improve the efficiency of the monitoring system and automatically extract valuable information. Lastly, we pointed out that the park could consider sending a Request For Information followed by a Request For Proposal, and select a qualified third party company for further development and maintenance.

Authorship

Sections	Primary Author	Editors
Sections	Primary Author	Editors
Abstract	Nicholas Hollander	Yicheng Yang, Mingzhang Zhu
Acknowledgements	Mingzhang Zhu	Nicholas Hollander, Mingzhang Zhu
Executive Summary	Nicholas Hollander	Yicheng Yang, Mingzhang Zhu
Introduction	Yicheng Yang, Mingzhang Zhu	Nicholas Hollander
Background 2.0	Mingzhang Zhu	Nicholas Hollander, Yicheng Yang
Background 2.1	Nicholas Hollander	Yicheng Yang, Mingzhang Zhu
Background 2.2	Nicholas Hollander, Mingzhang Zhu	Yicheng Yang
Background 2.3	Mingzhang Zhu	Nicholas Hollander, Yicheng Yang
Background 2.4	Yicheng Yang, Mingzhang Zhu	Nicholas Hollander
Methodology 3.0	Mingzhang Zhu	Nicholas Hollander, Yicheng Yang
Methodology 3.1	Mingzhang Zhu	Nicholas Hollander, Yicheng Yang
Methodology 3.2	Yicheng Yang	Nicholas Hollander, Mingzhang Zhu
Methodology 3.3	Nicholas Hollander, Mingzhang Zhu	Yicheng Yang
Methodology 3.4	Yicheng Yang	Nicholas Hollander, Mingzhang Zhu
Methodology 3.5	Yicheng Yang	Nicholas Hollander, Mingzhang Zhu
Results 4.0	All	All
Discussion 5.1	Yicheng Yang	Nicholas Hollander, Mingzhang Zhu
Discussion 5.2	Mingzhang Zhu	Nicholas Hollander, Yicheng Yang
Discussion 5.3	Mingzhang Zhu	Nicholas Hollander, Yicheng Yang
Recommendations 6.1	Mingzhang Zhu	Nicholas Hollander, Yicheng Yang
Recommendations 6.2	Yicheng Yang	Nicholas Hollander, Mingzhang Zhu
Recommendations 6.3	Nicholas Hollander	Yicheng Yang, Mingzhang Zhu
Conclusion	Mingzhang Zhu	Nicholas Hollander, Yicheng Yang
Reference	Yicheng Yang	Nicholas Hollander, Mingzhang Zhu
Appendix A	Nicholas Hollander	Yicheng Yang, Mingzhang Zhu
Appendix B	Nicholas Hollander	Yicheng Yang, Mingzhang Zhu
Appendix C	Mingzhang Zhu	Nicholas Hollander, Yicheng Yang
Appendix D	Yicheng Yang	Nicholas Hollander, Mingzhang Zhu
Appendix E	Yicheng Yang	Nicholas Hollander, Mingzhang Zhu

Table of contexts

List of Figures

List of Tables

1.0 Introduction

Visitor and traffic congestion is a major problem experienced throughout the entire National Park system. In 2018, approximately 318 million people visited national parks (National Park Service [NPS], 2019). Congestion within the parks is experienced in parking lots, hiking trails, visitor centers, and various major attractions. (NPS, 2018a).

Acadia National Park, located in Maine, also suffers from visitor and traffic congestion. In 2018, approximately 3.5 million people visited Acadia National park. In that year, the park ranked seventh on the most popular national parks list (National Geographic, 2019). Compared to most other National Parks, Acadia National Park is rather small in size, comprised of only 49,705 acres of land. The Park is ranked 47th out of 60 national parks by area (NPS, 2018b). However, it is one of the most popular national parks in the United States. As a result, visitor density in Acadia National Park is very high leading to the problem of visitor and traffic congestion. A study by Resource Systems Group shows that from August 1st, 2015, to September 29th, 2015, Acadia National Park was visually crowded every day and physically crowded 80% of the time. Visual crowding means there are at least eight people per views cape and physical crowding means there are at least fifteen people per views cape (Resource Systems Group, 2017).

The National Park Service released the Final Transportation Plan and accompanying Environmental Impact Statement (FTP/EIS) on March, 2019 for Acadia National Park, which aims at providing safe and efficient transportation to visitors and protect park resources and values. Some key actions including creating a reservation system, expanding the public transit service, and setting indicators and thresholds would be implemented to improve traffic conditions. Establishing a comprehensive monitoring system would be a critical component for providing references to those indicators and ensuring that the amount of visitor use will not exceed the maximum capacity.

The goal of this project was to develop a comprehensive webcam system for monitoring the road and parking lot conditions and sending back the images on websites. For park rangers, this system can help gather real-time information as an indicator of critical congestion. For visitors, they can

check the current and past road conditions as a reference for planning their trip. To achieve this goal, we created a solar-powered cellular static webcam and proved that this system can continuously work in the field with minimal maintenance. We also proved the concept of a GPS activated mobile webcam system which will automatically capture images at chosen GPS coordinates and automatically upload them to a web server. In addition, a proposal design was made to demonstrate the functions of a potential web page for the use of park rangers and visitors.

2.0 Background

Acadia National Park's Final Transportation Plan (FTP) will be implemented with the purpose of improving traffic and parking conditions while ensuring the quality of visitor experience and natural resources are well protected. This project sought to verify the feasibility of establishing a comprehensive monitoring system to fulfill the FTP while including recommendations for Acadia National Park management. A key background review that describes park conditions, traffic congestion, FTP of Acadia National Park, and related issues was performed in the following sections.

2.1 Description of the Park

Acadia National Park is located across Mount Desert Island, the Schoodic Peninsula, Isle Au Haut, and many smaller coastal islands around Bar Harbor, Maine, some only a few meters across. About 30,000 acres are located on Mount Desert Island, constituting 60 percent of the total area of the park. Originally established in 1916 as the Sieur de Monts National Monument by Woodrow Wilson, its purpose was to protect the natural beauty along the Atlantic shore of the United States. In 1929, the park was renamed to Acadia as a condition of a large private land donation on the Schoodic Peninsula. Figure 1 below shows the location and scale of Acadia National Park relative to Mount Desert Island and the State of Maine.

Figure 1. Location of Acadia National Park

Even though the park has received many land donations throughout history, today it still remains one of the smallest National Parks with an area of fewer than 50 thousand acres. Despite this, it is one of the ten most popular national parks, with the number of visitors increased by 58.8 percent since 2009 to a staggering 3.54 million with most attending between the months of June and September (NPS, 2019c). Visitors can enjoy recreational activities including viewing the scenery, hiking, camping, bicycling, and canoeing. Within the park, several locations including Cadillac Mountain, Sand Beach, and Jordan Pond make up the most popular attractions, drawing the bulk of visitors, and directly contributing to the congestion issues which currently plague the park (NPS, 2019).

2.2 Traffic Congestion and Safety Concern

As Acadia National Park has become more popular, the existing road infrastructure has begun to struggle with the number of visitors attending each year. Many of the roads, constructed as many as one hundred or more years ago have become overwhelmed with recreational vehicles, pedestrians, cyclists, the Island Explorer and tour busses. This poses a serious threat to visitor safety, as emergency vehicles can become encumbered while attempting to navigate through masses of illegally parked vehicles in route to the scene of an emergency site. The massive overcrowding of vehicles also detracts greatly from visitor experience, forcing visitors to wait for traffic, skip attractions, or park illegally.

As annual visitation rates have continued to rise over the past 28 years, the parks limited infrastructure has been forced to withstand 51% more traffic, contributing to an exponential increase in congestion (NPS, 2018a). In 2005, the park experienced just over two million visitors, however by 2018 that number had increased to nearly 3.6 million.

Previous research into the subject of traffic congestion in Acadia is plentiful, having been the topic of numerous academic analyses over the past 20 years. In “Preparing Acadia National Park for Modern Tourist Congestion”, it was determined that traffic congestion posed a serious issue during several peak time slots in which the number of visitors could increase by as much as 600% in less than an hour. (Cosmopulos, Gaulin, Jauris, Morisseau, &Quevillon, 2017)

One major contributing factor to the congestion is the intrinsic link between automobiles and the National Park Service. For many visitors to the park, "[Taking an] ocean Drive is important because it allows access to and enjoyment of the scenery" (Hallo & Manning, 2009). In total, 96% of surveyed respondents claimed that driving was an important part of the park experience (Hallo & Manning, 2009). With visitor traffic increasing at an ever greater rate, and the great American pastime of driving not predicted to go anywhere anytime soon, congestion is only projected to get worse unless something is done.

Cadillac Mountain is the most crowded point for visitors to Acadia National Park since visitors can enjoy magnificent views of dotted island landscape from all directions. Cadillac Summit is only accessible through a twisting and narrow road which is closed from December to April. Studies indicated that 75% of visitors went to Cadillac Mountain Area during their stay (Monz et al., 2010). Between June 2,2016 and October 31,2016, about 259,000 vehicles and 777,000 visitors went to Cadillac Summit (Charles Jacobi, 2016). The road to the summit was closed 70 times in 2017 due to congestion issues (Kong & Ring, 2019). Figure 2 below shows the traffic congestion of Cadillac Summit Road.

Figure 2. Congestion on Cadillac Summit Road (NPS, 2019b)

2.3 Final Transportation Plan

Since the congestion problems created safety issues, natural and historic resources protection concerns, and adverse impacts on visitor experience, the National Park Service (NPS) released the

Final Transportation Plan and accompanying Environmental Impact Statement (FTP & EIS) on March, 2019 for Acadia National Park (NPS, 2019a). The FTP aims at giving a comprehensive approach to provide safe and efficient transportation to visitors and protect park resources and values. The following are the key actions related to project.

Reservation Systems: One of the most critical actions undertaken by NPS in regards to traffic congestion is the reservation system. The purpose of the reservation system is to control visitor flow volume within the desired visitor capacities not including cyclists or pedestrians in popular scenic spots: Cadillac Summit, Thunder Hole, Jordan Pond, Sand Beach, Bubble Rock, etc. Reservations can be made online or through automatic service machines located at the Hulls Cove Visitor Center and the Acadia Gateway Center. The number of reservations or the duration of parking reservation for each location would be adjusted for each different time period to provide the best visitor experience by preventing traffic congestion and parking lot backups. If the volume of transit at specific location still exceeds set thresholds after the reservation systems for recreational vehicles are implemented, the reservation system could be extended to Island Explorer in order to manage traffic issues.

Public Transit: The Island Explorer service is a public transportation system with ten routes linking hotels, popular attractions (except Cadillac Summit), campgrounds, and visitor centers on Mount Desert Island and the Schoodic Peninsula (Downeast Transportation, 2019). Since the Island Explorer service cannot meet demands at peak volume of transit, increasing the number of Island Explorer buses becomes another important component of the Transportation Plan. The island explorer also does not travel to the summit of cadillac mountain, one of the parks most popular attractions and sources of traffic congestion. As described in the last section, Island Explorer service would also be expanded to cohere with the reservation systems requirements.

Visitor Information: The National Park Service plans to provide more varied and detailed information to visitors, both before and after they arrive at the park. That increased information includes reservation requirements and availability, trip-planning tools, orientation tips, advice on vehicle and bicycle safety, and car-free road sections. Information about traffic congestion and parking availability would be monitored and disseminated to the public, and it can be accessed

online or through mobile applications. Additionally, the park will improve cellular connectivity throughout the park by cooperating with communication providers. With better cellular signal, visitors have the capability to receive more real-time information from the park as a reference for planning their trips.

2.3.1 Existing Monitoring Strategies in the Plan

The National Park Service would adopt the Visitor Use Management Framework created by the Interagency Visitor Use Management Council (NPS, 2019a). This Framework includes establishing indicators, thresholds, and determining visitor capacity, in which monitoring is used to track these conditions over time (Interagency Visitor Use Management Council, 2016). According to monitoring results, park rangers could adjust the management strategies to ensure that the amount of visitor use will not exceed the maximum threshold. The following list below includes several Monitoring Strategies that would be implemented in the plan.

Vehicles at one time (VAOT): VAOT is a measurement technique commonly used by park rangers to quantify the amount of vehicles in parking lots and right-lane parking (Manning, Lawson, Newman, Hallo, & Monz, 2014). VAOT will be used as an indicator of transit and access conditions at crowded attractions (e.g., Thunder Hole, Jordan Pond, Sand Beach, Bubble Rock, Echo Lake Beach, Cadillac Summit). This indicator would also ensure the visitors with reservations of parking lots can arrive at their intended destinations as assigned.

People per Viewscape (PPV): PPV is a measure that helps park managers to quantify visitor crowding impacts along hiking trails, walking paths, and other scenic corridors in national parks (NPS, 2019a). PPV will be monitored on at least one of the following higher use trails: Jordan Pond Path, Beehive Trail, South Bubble Trail, Wonderland Trail, Cadillac Mountain Gorge path, Schoodic Head Trail (NPS, 2019a).

Automated vehicle traffic recorder (ATR): ATRs are devices for collecting large amounts of traffic volume data. ATRs can measure the direction of flow, traffic speed, and other crucial variables through pneumatic tubes (Accu-Traffic inc., 2010). Based on ATR data, NPS plans to

establish statistical and mathematical relationships with other indicators like VAOT, vehicles percent time following, and PPV for visitor use management.

2.3.2 Webcams as Additional Monitoring Strategies

The monitoring strategies which would be implemented in the Transportation Plan are all measurable attributes (e.g., VAOT, PPV) that can help park rangers to make changes in resources and conditions related to visitor experience. However, these data cannot . Therefore, webcams in various parking lots are more effective at determining current congestion situation. The real-time images been sending back from webcams can shorten the reaction time for appropriate managerial decisions.

2.3.3 Webcams as Information Provider

The Nation Park Service currently provides some basic information (operating hours, weather, closure conditions, fees, park regulations, etc.) to visitors for planning their trips. As described in the 'Visitor information' section of the Plan, the National Park Service sought to provide more information about congestion and parking availability and enhanced trip-planning tools to visitors. Webcams are a great information source by uploading real-time images to the public website so that these images can be accessible to the public. With better cellular connection in the future, visitors could browse the conditions of road and parking lots of all the attractions upon arrival so that visitors can adjust their schedule to avoid traffic congestion. By providing this information, visitors would have a more comprehensive picture of the transit conditions of the park by browsing images of previous years before they arrive at the park.

2.4 Related Project Overview

2.4.1 Previous WPI Projects

Our project is an extension of the 2018 Acadia Webcam IQP. In 2018, they explored the "Feasibility of Webcam Implementation in Acadia National Park". In this project they

benchmarked the webcams used in Yellowstone National Park and decided to experiment with webcams as a solution for the traffic congestion situation in Acadia National Park. In order to prove this idea is feasible, they also did research on the cost of webcams as well as the visitor rights and ethics associated with surveillance technology. It turned out that webcams were not expensive and monitoring in the park was unlikely to be a violation of privacy rights. Thus, they utilized the Spypoint Link-S camera to take pictures in fourteen locations throughout the park and successfully got decent pictures. After determining the camera's battery performance, cell signal and picture quality, they concluded that "a webcam could be a reliable tool to remotely monitor traffic in parking lots" (Bruno, Cromwick, &Feng, 2018). They also suggested the connectivity and power source might be a limitation that need to be worked on.

Besides the 2018 Acadia Webcam Team, this project also took the works from 2018 Acadia Cellular Connectivity Team and 2018 Glacier Congestion Management Team as references. The 2018 Acadia Cellular Connectivity Team measured cellular connectivity status in Acadia National Park. They recorded the cellular signal strength and internet speed of different providers throughout the island. The final cellular signal heat maps suggested AT&T is the most reliable and consistent signal provider on Mount Desert Island (Bergquist, Palacios, &Goklevent, 2018). Additionally, the 2018 Glacier Webcam Team explored the feasibility using a Raspberry Pi as the platform for the webcams. They also demonstrated the feasibility of using File Transfer Protocol over a cellular connection (Barrameda, Vose, & Rizzo, 2018). This allowed the pictures taken by the webcams upload to a website directly.

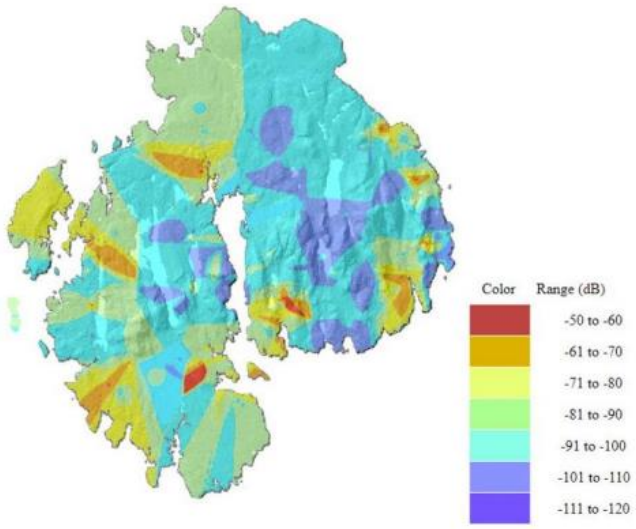


Figure 3. AT&T Cellular Signal Heat Map (Bergquist, Palacios, &Goklevent, 2018)

2.4.2 Existing Cameras in Acadia National Park

At present, only one webcam placed at North Atlantic Coastline is working in Acadia National Park (as seen in Figure 4). This camera funded by Canon aims at "Offering glimpses of sunrise, storms, sailboats, and the occasional seagull" (NPS, 2018c). A static image is taken every minute and published online through cellular data connectivity by daylight. The Camera is powered by a rechargeable battery and a solar panel, which ensures that the camera can work continuously.

Figure 4. North Atlantic Coastline Webcam

3.0 Methodology

The goal of the project was to develop a comprehensive webcam system for monitoring the road conditions and uploading images to the cloud. For park rangers, this system can help gather real-time information as an indicator of critical congestion. For visitors, they can check the current and past road conditions as a reference for planning their trip. This project contains three main components: a static webcam intended for installation at a fixed location, a mobile webcam intended for installation on a vehicle such as a bus, and mockup websites. To achieve this goal, five subsidiary objectives were accomplished:

Objective 1: Design a static webcam system

Objective 2: Implement a static webcam system

Objective 3: Verify the concept of mobile webcam system

Objective 4: Make a mock-up of web pages that will display the images from webcams

Objective 5: Evaluate the data and offer recommendations

3.1 Objective 1: Design a Static Webcam System

This section describes the design goals, the process and analysis of how we determined the appropriate components of the system, how these components work together, and the cost of the system.

3.1.1 System Design Principles

This static webcam system seeks to achieve the following goals:

- Operate continuously for months with low maintenance in all weather conditions.
- Able to upload real-time, high-resolution images to public websites periodically.
- Self-contained power supply without grid connection
- Low cost compared to professional security camera
- Moderate size, easy to installed

3.1.2 Hardware Selection

In the beginning, we considered using the Spypoint Link-S, the one used by the webcam team last year to determine the feasibility of traffic monitoring in Acadia National Park. The SkyPoint Link-S is a solar cellular trail camera with supporting mobile applications and private cloud storage (Bruno, Cromwick, &Feng, 2018). With these accessories, users can easily set up the webcams and put them into use. However, it is not possible for us to make modifications and add components as needed since the Spypoint Link-S is not an open source system. Additionally, users have to pay monthly fees to get accounts to view the images. This means that the accessibility to those images is very limited because park rangers and the public would have to use one account. For these reasons, we decided not to continue with last year's setup.

Figure 5. Spypoint Link S Camera

We chose the Raspberry Pi, a credit-card sized single board computer, as our central processor. Flexibility is the major advantage of this setup compared to the SkyPoint Link-S and other existing products. Since the Raspberry Pi runs a suite of open source software, it is easy to program as we need. With the matched camera module, we can adjust the size of the image and the time interval based on actual situations. The LTE (Long Term Evolution, a standard for 4G wireless broadband communication for mobile devices and data terminals) module gives SFTP compatibility to Raspberry Pi, which allows the images to be uploaded to public websites.

The power supply for the webcam system contains a battery and a solar panel used to charge the battery. We selected SLA1116 to be our test battery (12V, 18Ah). With this battery, the system could last approximately two and a half days without solar panel. Please refer to "Appendix A: Battery Estimation" for detailed calculations based on the power consumption. The 50W solar panel we chose is twice the power required so that the battery can be fully charged in a short period of time.

A transparent weatherproof box is used to prevent electrical components from water and moisture. The battery and solar panel are also waterproof for the purpose of surviving in the field. When we were doing some tests prior to setting-up the device in the field, we noticed that some water drops could hang on the acrylic surface on rainy or foggy days. A hydrophobic coating spray was adapted to solve this issue while ensuring the camera's view would not be blocked. The weatherproof box is shown in Figure 6 below.

Figure 6. Weatherproof Box

3.1.3 System Structures

Figure 7 shown below introduces the main compositions and working principles of the static webcam system. As the central processor of the system, the Raspberry Pi is connected to the LTE module (EC-25A), buck converter, and camera. The LTE Antenna helps the system achieve better cellular connection at the summit of the mountain. The Buck converter is used to convert the 12V

input voltage from the solar charge controller to 5V, the normal operating voltage of the Raspberry Pi. The Buck converter is connected to the solar controller, a small box consisting of solid state circuitry. Its function is to regulate the amount of charge from the solar panel so that the batteries will not be over or under charged. The weatherproof box contains all the components except the LTE antenna, solar panel, and battery. Figure 8 below shows key components of the system.

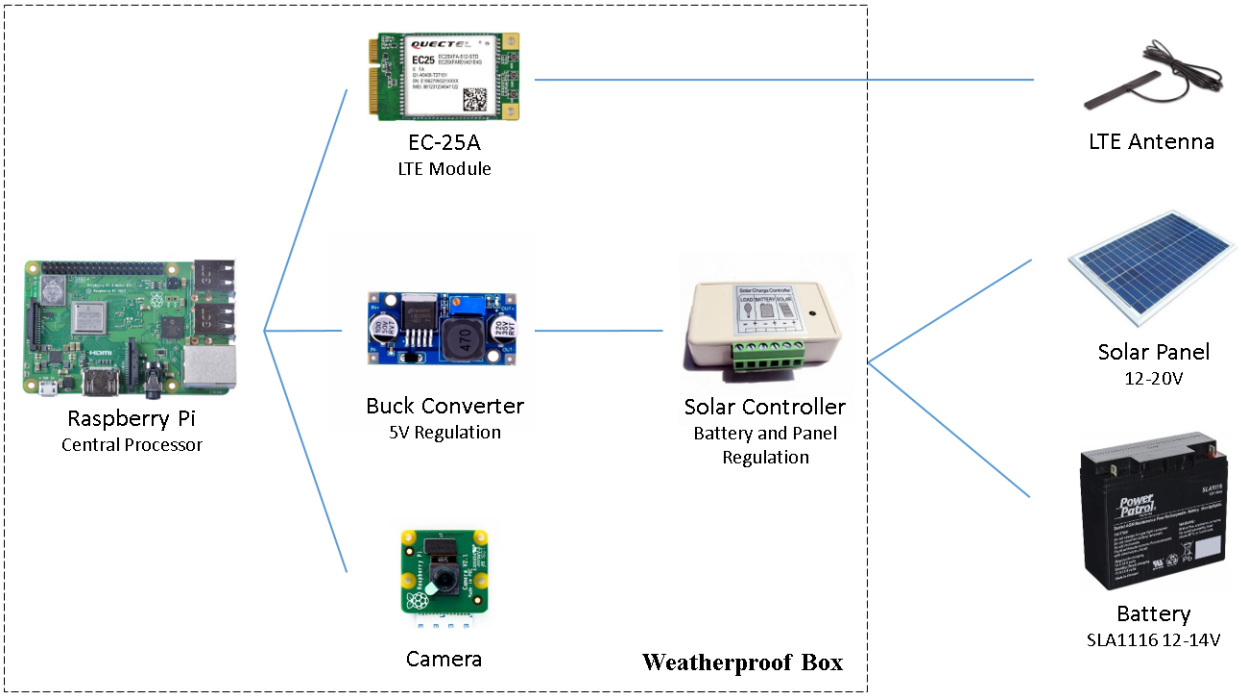


Figure 7. Static Webcam System Composition

Figure 8. Components of Static Webcam System

3.1.4 Cost of the Static Webcam

The total cost of the hardware components is about $350, which is relatively low compared with existing outdoor security cameras. These cameras usually cost around $600 to $1000. Additionally, the data plan carries a fee of approximately $50 a month to upload the images to the server. Please refer to “Appendix B: Static Camera Bill of Materials” for a detailed list of components including the quantity and price.

3.2 Objective 2: Implement a Static Webcam System

We put the camera at the Blue Hill Overlook on Cadillac Mountain. The reason for that is the serious over-crowded conditions on Cadillac Mountain. The summit road to Cadillac Mountain was closed 54 times in 2018 due to traffic congestion (Kong & Ring, 2019). By putting the camera near Blue Hill Overlook, we can not only see the traffic conditions at the entrance of Blue Hill Overlook parking lot, but also monitor the road for vehicular congestion back-up from the summit of Cadillac Mountain. On the other hand, the traffic conditions near the exit of Blue Hill Overlook is an important indicator to the summit of Cadillac Mountain. Occurrence of traffic congestion around this area indicates that the Cadillac Summit could exceed its visitor capacity, and the Cadillac Summit Road should be closed.

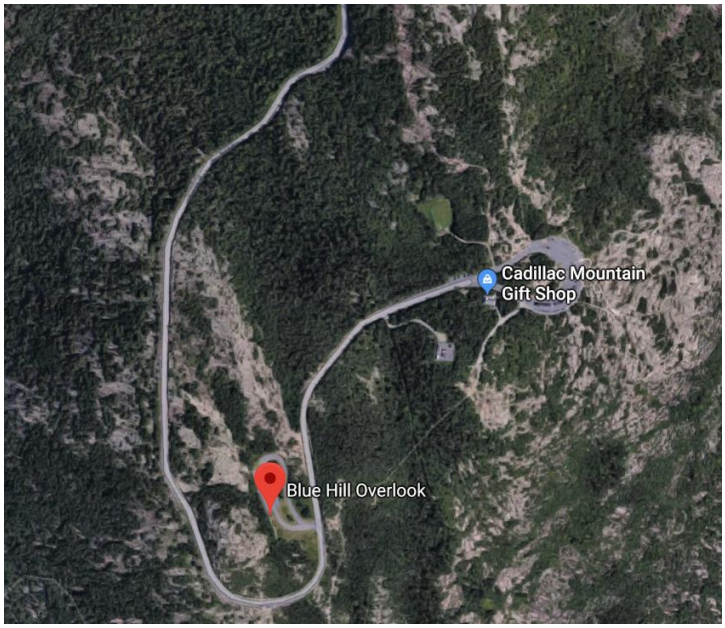


Figure 9. Location of Blue Hill Overlook (Google, 2019)

To achieve the goal of monitoring, a location with a wide view was required. Therefore, we did a field investigation. There were some high trees we believe might be the ideal location for the camera, but we had to consider the safety and equipment limitations. We finally decided on a relatively low tree which provided an acceptable view. Figure 10 shows the exact location of the static webcam. The camera itself was fastened to the trunk through two bungee cords. The solar

panel was placed on branches and reinforced by nylon rope and studdles. The battery was semi-buried in the soil under the tree to protect it from environmental conditions. The final setup of our system is shown in Figure 11 below.

Figure 10. Location of Static Webcam System

Figure 11. Setup of Static Webcam System

The webcam was programmed to take pictures from 4 a.m. to 8 p.m. In this way, the webcam could monitor from early morning and would not miss the crowd of people who came for sunrise or sunset. To transfer the images, a data plan of 18G from AT&T was bought and installed on the webcam. The time interval between two pictures was set to two minutes to optimize power and data usage. This number could be easily adjusted if necessary. Once the camera captured an image, it would automatically upload the image to a server that can be viewed online. The Figure shown below is an example photo taken by static webcam.

Figure 12. Example Image taken by Static Webcam

3.3 Objective 3: Verify the concept of a mobile webcam system

3.3.1 Design a Mobile Webcam System

After development and implementation of the static webcam platform was complete, we began investigating the possibility of using a GPS activated mobile camera platform for capturing images at a variety of locations. To do this, we utilized the Mobile Capture Platform developed in conjunction with and constructed by students in the Glacier National Park Webcam Team.

When developing the platform, we were faced with four major challenges. Capturing the images, processing the images, transmitting the images, and preparing the images for consumption. We first investigated a number of commercially available solutions such as the Spypoint camera

utilized by the 2018 Acadia National Park Webcam Team, however there were a number of limitations to the platform that we wished to avoid. The most significant limitation was the inability to utilize a custom image hosting solution. This made it impossible for multiple users to access data simultaneously, and significantly complicated the process of viewing historical trends. Another issue was the reduced image quality resulting from the low-power image sensor.

We ultimately settled on a modular system built around the inexpensive Raspberry Pi 3 computing platform. The Raspberry Pi platform includes a high-resolution external sensor, which allowed high quality images to be captured, a feature absent from many commercial solutions we evaluated. The onboard processor allowed us to perform important operations on the images, such as annotating identifying information, and compressing the images to reduce bandwidth.

Transmission of images proved to be one of the most challenging aspects of this objective. Many camera platforms in use today rely on WiFi or Ethernet, neither of which have a range exceeding 100 meters. While some cameras do feature built in 3G connectivity, a growing number of cellular service providers have announced plans to shut down their 3G networks, some as soon as 2019.

Figure 13 shown below is the system composition of our mobile webcam system. The setup is pretty similar to the static webcam. As the central processor of our system, a Raspberry Pi is connected to a camera module and EC-25A, a GPS and LTE module. These components are contained in a weatherproof box. LTE Antenna and GPS Antenna is connected to the EC-25A for boosting the signal. The system is simply powered by a USB car charger. Figure 14 shows the real products of the mobile webcam system.

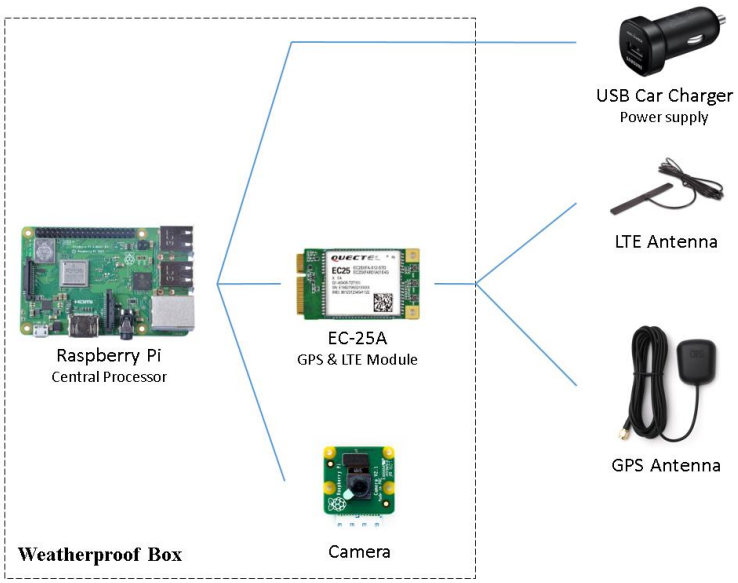


Figure 13. Mobile Webcam System Composition

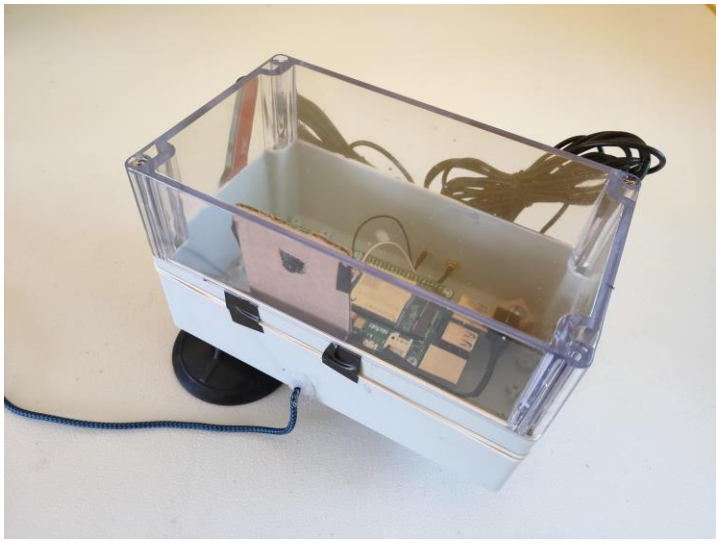


Figure 14. Real Products of Mobile Webcam System

3.3.2 Implement a Mobile Webcam System

The mobile webcam was mounted on the top of our private vehicle through two suction cups while testing the feasibility and reliability of our system. Figure 15 shown below is the setup of the system. However, it was designed to be implemented on the Island Explorer Service, a public transportation system with ten routes linking hotels, attractions, campgrounds and visitor center on Mount Desert Island. We chose Loop Road (one of Island Explorer's routes) as our simulated

route since it contains nearly all the most crowded destinations in the park (Visitor Center, Sieur de Monts, Sand beach, Thunder Hole, Jordan Pond).

Figure 15. Setup of Mobile Webcam System

To determine the locations of the places that needed to be monitored, we used the Park Loop Road and the Island Explorer route. We recorded 32 locations along the Park Loop Road route. These locations included road intersections, parking lots and entrance, trailheads, Hulls Cove Visitor Center, and Carriages of Acadia. Figure 16 below is the comparison of the Park Loop Road route and the GPS locations where we chose to capture images. Since the Island Explorer, and our testing vehicles, would not pass exactly the same point every time, we set a configurable radius of 10 meters for each coordinate. Once the vehicle entered this range, the camera would capture an image. Those images were stored in the internal flash drive and uploaded to the cloud once there was connectivity.

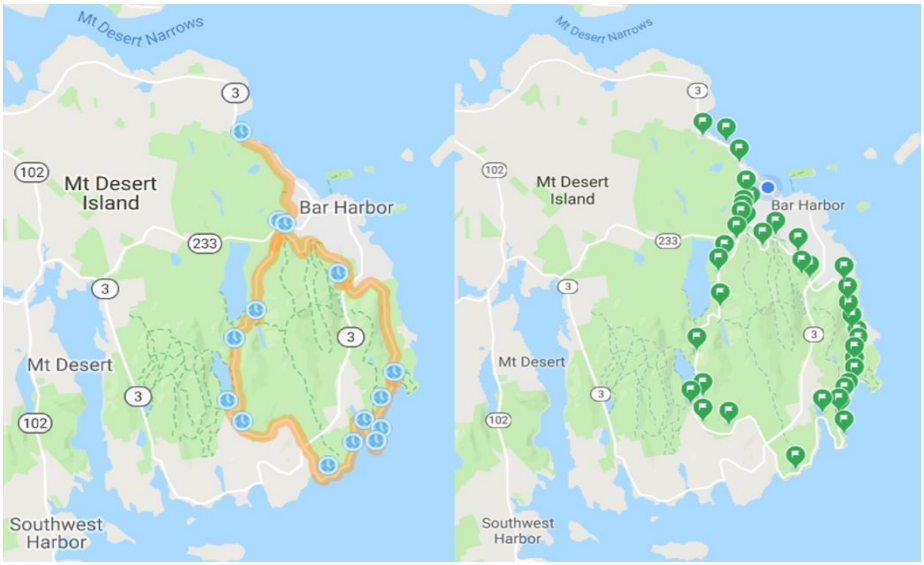


Figure 16. Loop Road Route vs. GPS Location Selected

3.4 Objective 4: Create a Prototype Web Page to Display the Images from the Webcam

The purpose of the web page was to display, for the rangers and visitor's use, the images captured by the webcam. Because the web page would eventually contain images from multiple webcams positioned at parking lots throughout the park, the web page was designed to simulate a future webcam implementation. The web page needed to be simple and clear, especially for those visitors unfamiliar with Acadia National Park.

The goal was to create two web pages: one to represent *Current Conditions* and the other to represent *Past Conditions*. Thus, there would be two similar but independent web pages. The prototype web pages, though not functional online web pages, were realistic and could be easily integrated into the official Acadia National Park website in the future.

Figure 17. Web Header from Acadia National Park Official Website

The next things added to both web pages is route and title. The size, font and color of these characters were exactly the same as the ones on the official website. (NPS, 2019b)

Figure 18. Header of Prototype Web Page

Blow the title, the first thing users would see was a calendar. On the web page for *Current Conditions*, the calendar would show only the current date. But, on the web page for *Past Conditions*, the calendar would be clickable and users could choose a date from the past.

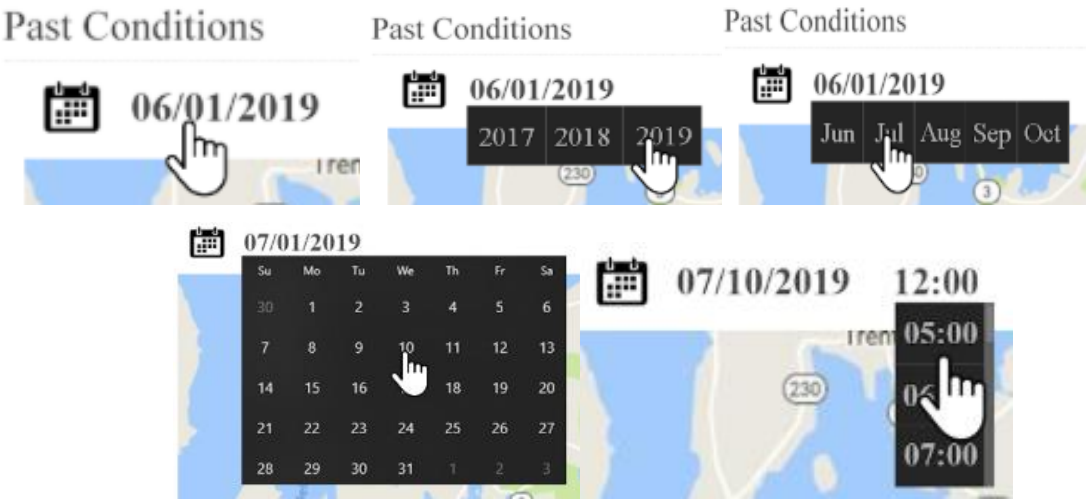


Figure 19. Example of Choosing Date & Time

In addition to the drop-down calendar navigation, an Acadia map is displayed. There are 15 icons on the map, representing the 15 primary parking lot locations in the park. On the left of the map appears a search box that includes the names of the parking lot locations. This was important since new visitors may not be familiar with the parking lot locations. The search box could be closed and removed from view if necessary.

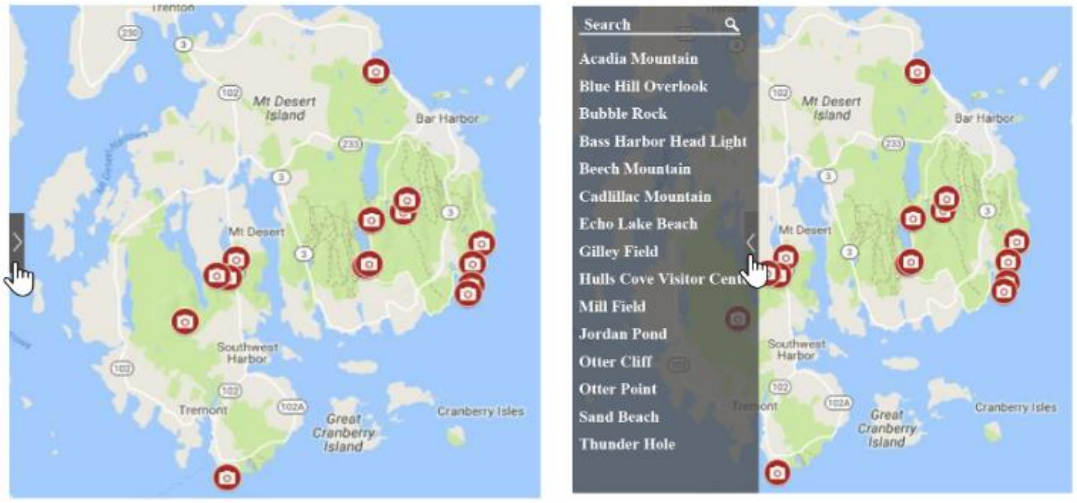


Figure 20. Search Box Hidden vs. Search Box Shown

Whenever users hover over a parking lot camera icon on the map, the name of that parking lot will pop up. This would help users to confirm a location. When users click on the icons, the images of that parking lot location would appear. On the top-left of the images would be the name of that parking lot location. On the top-right side would be the time when the photo was taken. For the *Current Conditions* web page, the images would be presented in real-time, and the number of images would be determined based on number of webcams at the specific parking lot location. For the *Past Conditions* web page, all the images of the chosen date would show up as a horizontal list. There would be 15 images representing the conditions hourly from 5 a.m. to 8 p.m.

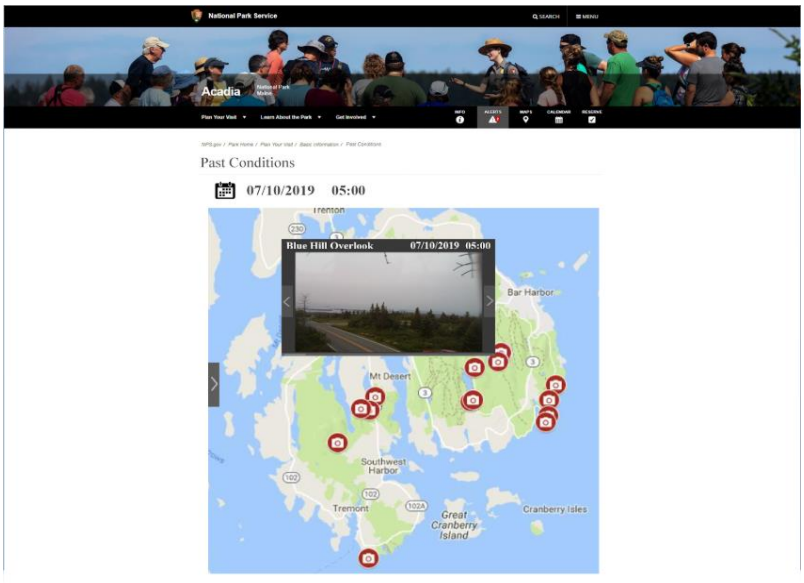


Figure 21. Example of Viewing Past Conditions

The images used are relatively small (thumb nail) when users first view them. But, they could be enlarged by clicking the images. The reason not to initially enlarge the images was based on allowing the users to view the entire page without obstruction. In this way, those who want to learn the traffic conditions roughly can view the images efficiently, and those who want to view more detail need to click the image one more time.

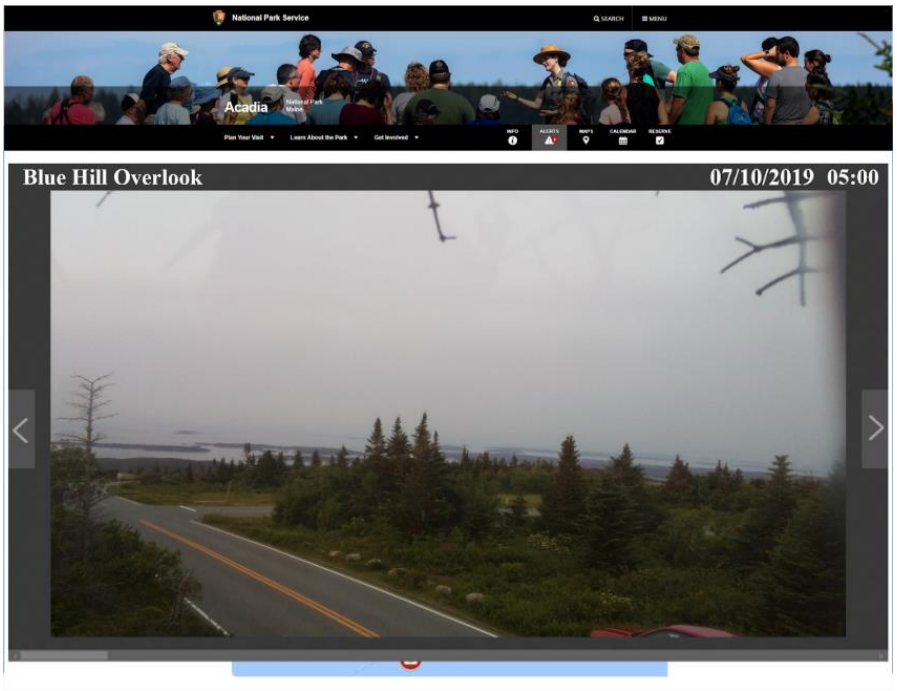


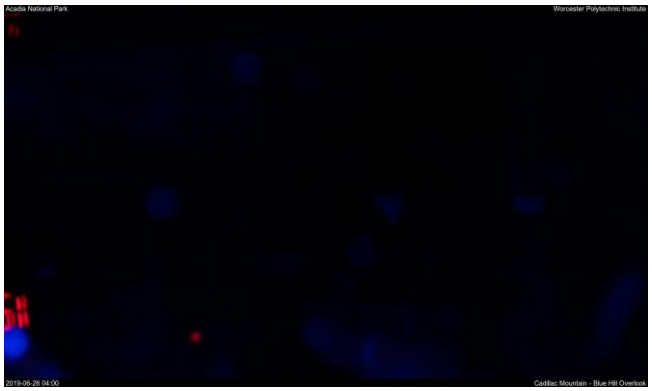
Figure 22. Example of Enlarging an Image

3.5 Objective 5: Evaluate the Data and Offer Recommendations

The purpose of this objective was to analyze the webcam systems for problems or limitations. There were two aspects that needed to be evaluated: image quality and battery life.

First, for both kinds of webcams, image quality would be the most important data to evaluated. For the static webcam system, we decided on a 1-5 point standard based on how clear the image was.

Score	Description	Example

1	Nothing can be observed	
2	Limited view which excludes entrance of Blue Hill Overlook and the curved road.	
3	Limited view which includes part of entrance of Blue Hill Overlook but excludes the curved road.	

4	Clear view until the entrance of Blue Hill Overlook but excludes the curved road.	
5	Clear view with the whole path	

Table 1. Image Quality Grading Standard

By giving a score to every image based on this standard, we could arrive at a percentage score for each image.

Second, the durability includes the weatherproof ability and the battery life of the webcam system. This would only be applied to the static webcam since the mobile webcam is powered by the car battery. The way to evaluate the durability would be a daily check. The voltage of the battery would be recorded and team members would also check the conditions of the camera box, the solar panel, and the battery. This information would reflect the durability of the webcam and how long it can operate continuously.

4.0 Results

Date	Sky coverage at different timestamp (%)						Voltage (V)
	5am	*8am*	*11am*	*2pm*	*5pm*	*8pm*	
4-Jul	18	16	14	14	12	12	14
5-Jul	26	24	20	18	24	29	14
6-Jul	42	56	64	78	52	29	14.4
7-Jul	6	6	4	4	4	4	14.2
8-Jul	6	6	4	1	1	1	13.8
9-Jul	4	4	4	15	13	24	13.9
10-Jul	26	37	10	29	45	62	14
11-Jul	46	63	76	76	77	79	13.6
12-Jul	100	100	100	96	89	85	12.8
13-Jul	82	78	66	53	51	21	13.8
14-Jul	72	76	47	50	49	37	13.8
15-Jul	41	47	32	35	22	18	13.8
16-Jul	26	19	12	17	27	47	13.8
17-Jul	69	77	88	96	98	91	13.2
18-Jul	26	16	15	28	23	10	13.7
19-Jul	70	23	24	37	49	38	13.9
20-Jul	77	49	45	49	49	56	14.4
21-Jul	72	60	67	67	57	41	13.9
22-Jul	56	42	52	61	78	95	13.8
23-Jul	94	100	95	90	88	56	13.5

Table 2. Voltage with Sky Coverage at Different Timestamp

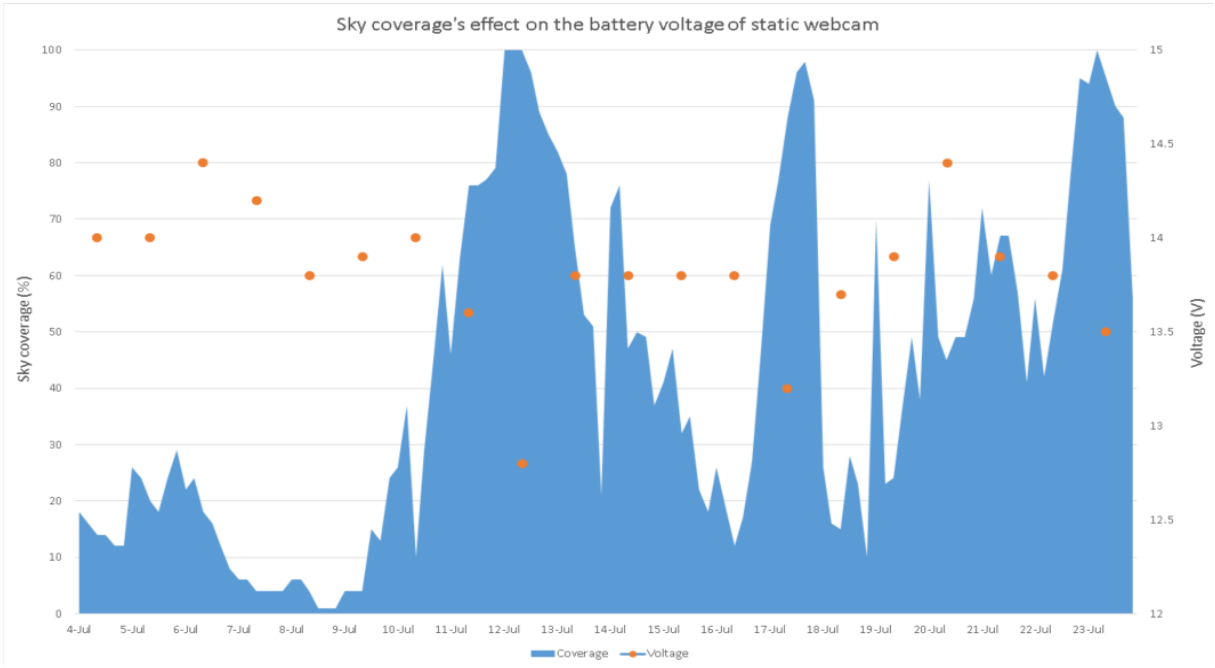


Figure 23. Sky Coverage vs. Battery Voltage

The static webcam system continuously worked for 20 days, from July 4th to July 23rd without any maintenance. We adapted sky coverage as an indicator of how much power the battery required to be charged by the solar panel. We recorded the sky coverage percentage six times a day (National Oceanic and Atmospheric Administration, 2019). We also went to the Blue Hill Overlook every day at about 11:00am to check and record the battery voltage.

Table 2 and Figure 23 above show the Sky coverage versus battery voltage. The normal working voltage of the static webcam is about 13.8v - 14.0v. It was found that when the sky coverage reaches up to 100 percent (Jul 12nd, Jul 17th, Jul 23rd), the battery voltage was comparatively low ranging from 12.8v to 13.5v. However, it was still enough to keep the system working. The webcam would shut down if the voltage dropped lower than 10.8v. Also, as long as the sky coverage didn't drop below 50 - 60 percent, the battery would be charged up to 13.8v, which proved that the existing solar panel had the capability to charge the battery and keep the system working during continuous cloudy days.

Date	1	2	3	4	5	Total	Note
07/04/2019	0	0	0	0	453	453	
07/05/2019	0	0	0	0	454	454	
07/06/2019	0	0	0	41	411	452	
07/07/2019	0	0	0	0	148	148	Server died for half a day
07/08/2019	0	0	0	0	456	456	
07/09/2019	0	0	0	0	455	455	
07/10/2019	0	0	0	0	455	455	
07/11/2019	0	0	0	0	456	456	
07/12/2019	29	282	113	27	5	456	
07/13/2019	15	5	54	9	373	456	
07/14/2019	0	0	0	1	454	455	
07/15/2019	0	0	0	0	395	395	Server died for half an hour
07/16/2019	0	0	0	0	455	455	
07/17/2019	0	10	28	10	407	455	
07/18/2019	0	0	0	0	455	455	
07/19/2019	0	0	0	0	455	455	
07/20/2019	0	0	0	0	455	455	
07/21/2019	0	0	0	0	455	455	
07/22/2019	0	0	0	0	456	456	
07/23/2019	0	0	0	0	362	362	Camera was taken off in the afternoon
Total Image	44	297	195	88	8015	8639	
Percentage	0.51%	3.44%	2.26%	1.02%	92.78%	100%	

Table 3. Image quality estimation of static webcam

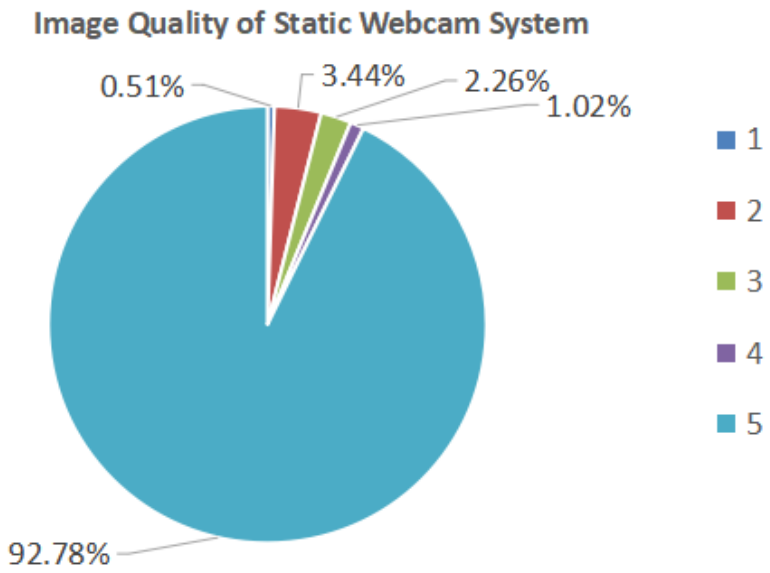


Figure 24. Image Quality of Static Webcam System

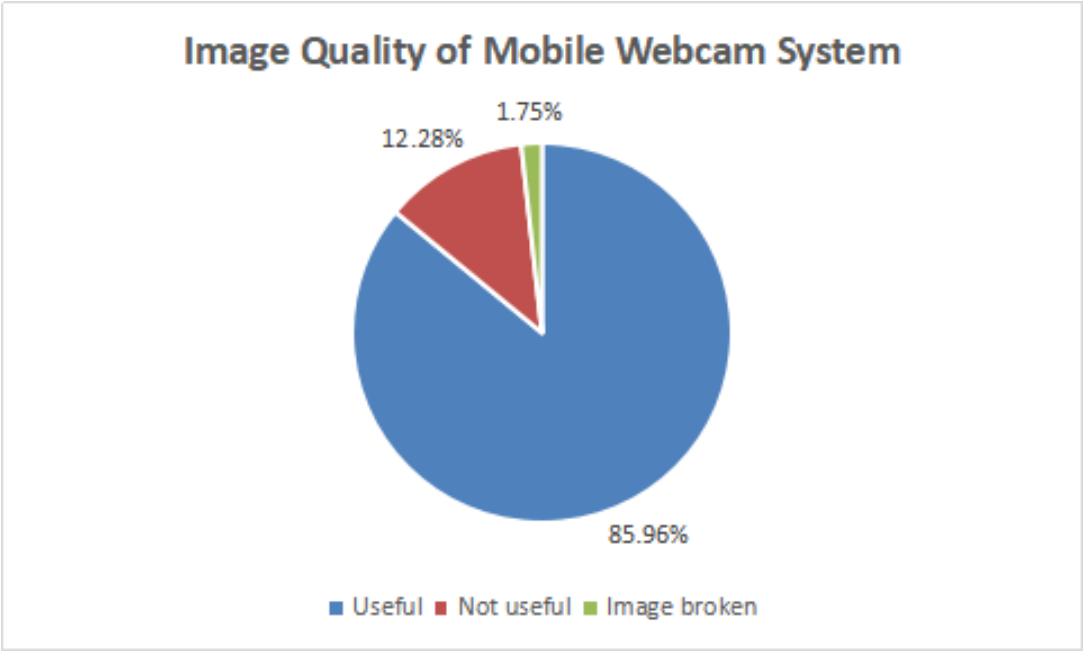


Figure 25. Image Quality of Mobile Webcam System

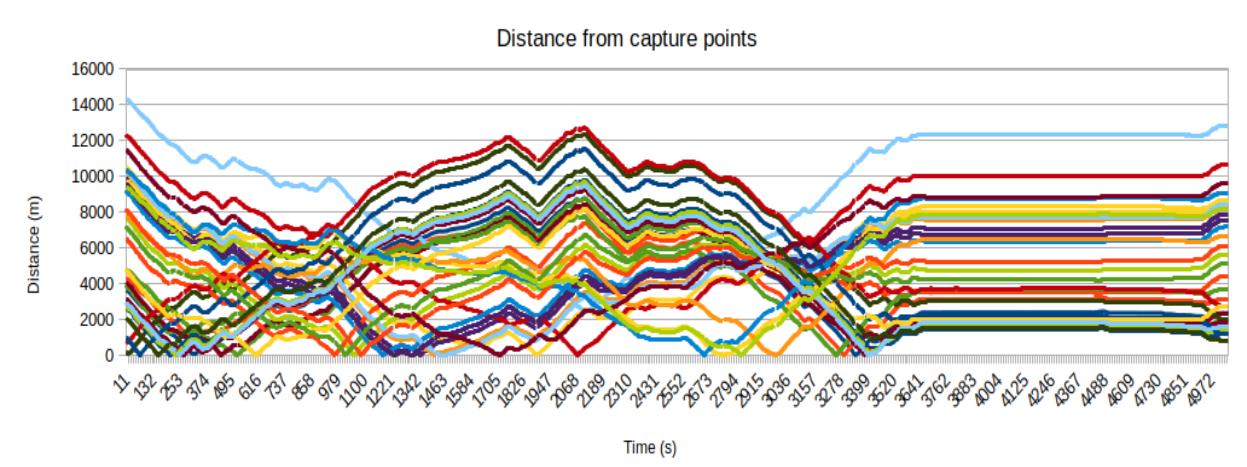


Figure 26. Distance to Capture Points

In Figure 26, you can see the distance between points as generated during a test drive around the park loop road. Each colored line represents the distance in meters between the test vehicle and the center of the trigger region. At each time where the measured distance is less than 10 meters, the camera is automatically triggered and an image is uploaded.

5.0 Discussion

5.1 Image Quality

Overall, the image quality for both webcam systems was good. For the static webcam system, about 92.78% of the images taken were considered 5 points, which meant the whole path through Blue Hill Overlook could be viewed in these pictures. About 4% of the images were considered 1 or 2 points. The reason for that was the foggy weather. The huge fog decreased the visibility which also decreased the image quality. This is a problem that is not easy to solve. On the other hand, for a regular rainy day, the score of the pictures are usually above 3 points since we had applied hydrophobic coating spray on the box so that the raindrops would not stay on the shield.

Figure 27. Image under Different Weather Conditions

For the mobile webcam system, 85.96% of the images are considered useful which means these images can either clearly reflect the traffic conditions on the roads or in the parking lots. The reason that 12.28% of images are considered not useful is the GPS location. Since it is difficult to drive through the exact same point every time, we added a tolerance to the GPS coordinate. This would make the camera shoot the photo once the car entered that range. So some of these images which originally aimed to be shot after a curve were actually shot before the curve. We believe this problem could be fixed by taking more tests and decide on the tolerant range and algorithm to take the pictures.

Figure 28. Useful Image vs. Not Useful Image

5.2 System Size

Since the static and mobile webcams are prototypes and used primarily for functional and technical verification, the size of two systems are comparatively large. However, as the project proceeds and reaches a third party development company, like SPYPOINT or Bushnell, these companies would utilize advanced solutions to integrate all the electrical components. This would result in a much smaller camera and overall footprint. Since there is no battery or solar panel for mobile webcam, the final product could be as small as a GoPro and attached to the inside of the front window similar to a dashcam.

5.3 Webcam as an Indicator

As we described in Objective 2, the traffic conditions near the exit of Blue Hill Overlook could be used as an indicator of overcrowding at the Cadillac Summit. We cross-checked the images captured at the Blue Hill Overlook to see if the images correlated with road closures. Some interesting patterns began to occur.

July 5th was an especially crowded day and the Cadillac Summit Road closed two times according to park management statistics. On July 6th, the webcam captured severe traffic congestion and illegal parking alongside of Cadillac Summit Road at 4:45am in the morning. However, the park did not take any actions at that time.

The same situation occurred on July 22th. The road became pretty congested and some dangerous conditions prevailed due to illegal parking. Based on the comparison result, we recommended that park rangers should pay attention to the traffic flow around Blue Hill Overlook especially during sunrise in the morning. Please refer to "Appendix C: Acadia National Park Closures Statistics from July 1st to July 22nd" for detailed information.

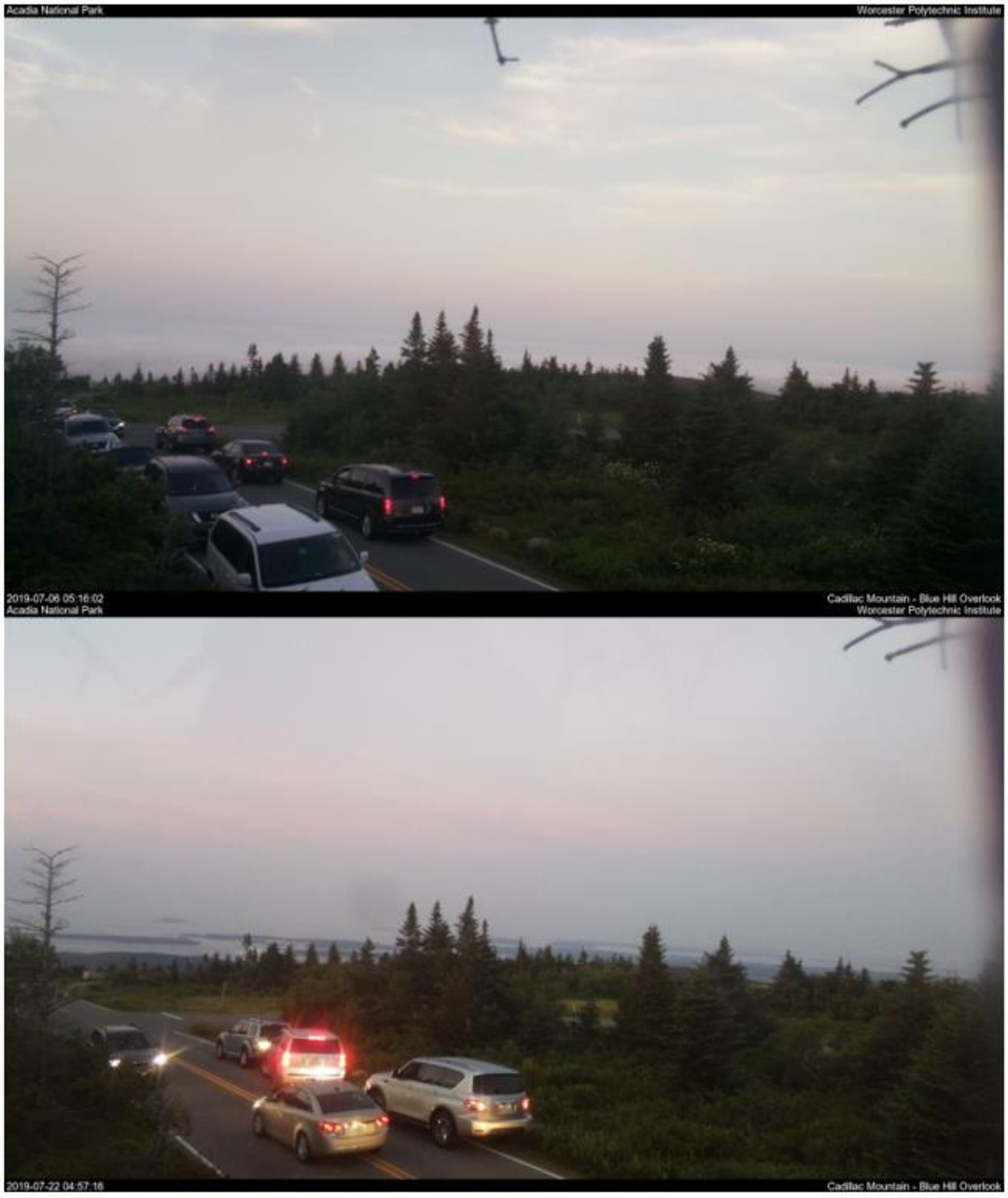


Figure 29. Congestion on Jul 6th and Jul 22th

6.0 Recommendations

6.1 Equipment Improvement

We did a comprehensive evaluation (cost, continuity, installability, potential problems, etc.) for both webcam systems. Several recommendations including battery capacity, transmission type, system size, and rebooting function was performed in the following sections.

6.1.1 Battery Capacity

As we described in the 'Hardware Selection' section in the Objective 3, the system can continuously work for about two and a half days if the solar panel is completely not working. This implies the power supply setup can survive without direct sunlight for about 3-4 days. Please refer to "Appendix A: Battery Estimation" for detailed calculations based on the power consumption. However, if extreme weather conditions occur such as raining continuously for an entire week, the system could power down.

We investigated the weather conditions from June to September for the past three years. The longest wet weather appeared from Jun 28th to July 3rd, which lasted for six days (timeanddate, 2019). Based on that information, we recommend that the battery capacity should be twice the size than the one currently used. The figure shown below is the battery PS-12350NB, a type of battery with suitable capacity (12V, 35Ah).

Figure 30. PS-12350NB Battery

6.1.2 Transmission Type

Both the static and mobile webcams used a cellular network connection for uploading data to the Internet, which requires a SIM card with a data plan for each device. If dozens of cameras are employed, a large amount of expense would be generated monthly due to data usage, so we recommended that in the future, a point-to-point Ethernet bridge network could be established for data transmissions.

This type of Ethernet networks involves a single transmitter and a single receiver. Point-to-point wireless can reach up to 10 miles if there is nothing blocks between two transceivers. The main strength of this transmission type is the reduction of recurring costs since once transceivers are properly installed and powered up, the connectivity can be established stably. Static webcam systems are suitable for this type of transmission. The Figure 31 is transceivers schematic.

Figure 31. Transceivers Schematic

We used airLink, an online transmission simulation software, to determine the feasibility of implementing this technology in Acadia National Park. We assumed that receivers are placed on an antenna 30 feet above the ground at Park Headquarters. Three transmitters would be placed separately on the Cadillac Summit, Blue Hill Overlook and Bubbles Divide Trailhead, 15 feet above the ground. The simulation result and signal strength for each transmission is shown in Figure 32 and 33.

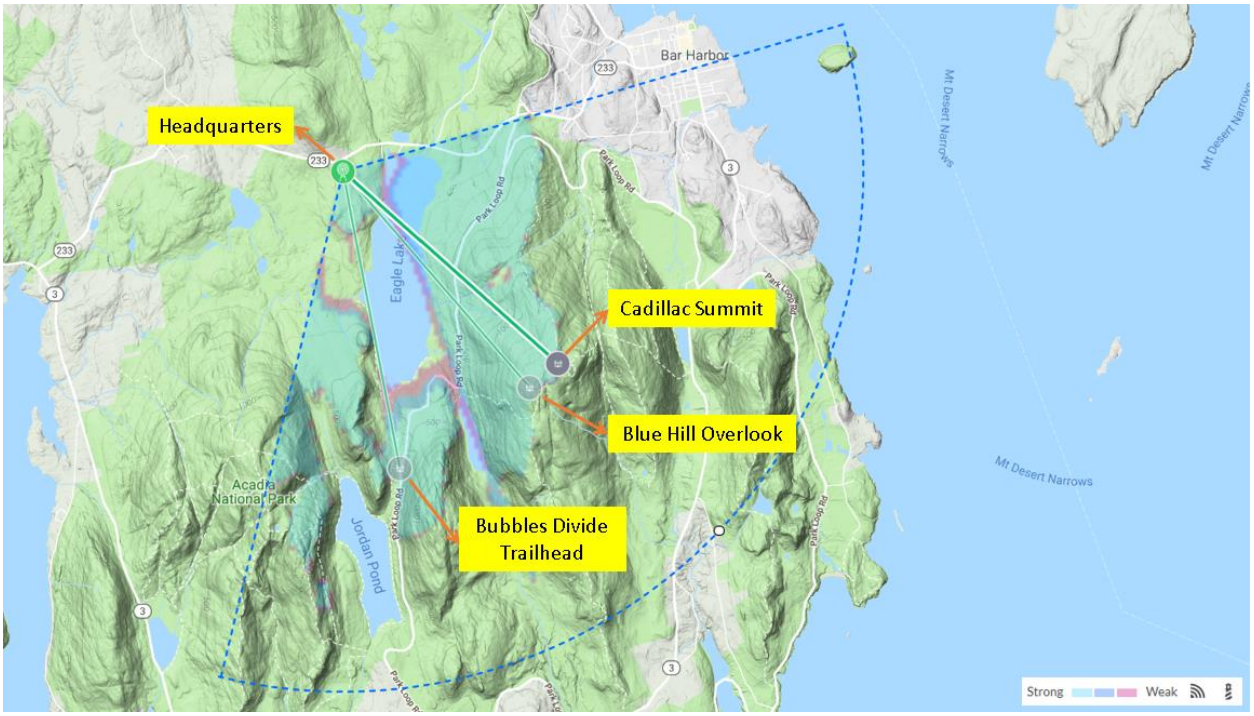


Figure 32. Transmission Simulation

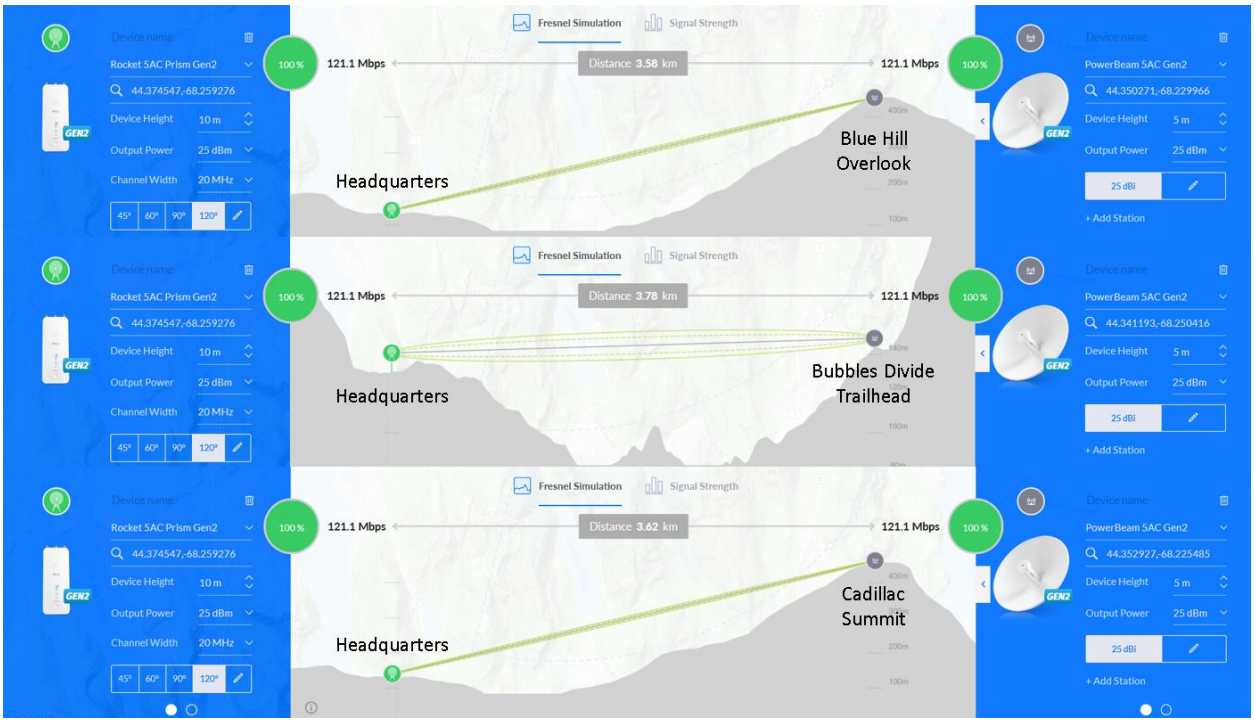


Figure 33. Signal Strength of Each Transmission

Based on the simulation, it can be found that the signal strength between those three places and Park Headquarters are pretty good. However, due to terrain effects, the area alongside Eagle Lake, as well as Jordan Pond and Sand Beach, cannot establish direct connectivity with Park Headquarters in this setup. Implementing a point-to-point Ethernet network including several receivers and transmitters could be a solution to get connectivity for all the locations in the park with a long-term plan.

6.1.3 Rebooting Function

Currently, our static webcam system will enter idle mode when it is not working, which will still need about 1.9W to keep Raspberry Pi alive. For a long-term use, we recommended adding a rebooting function so that the Raspberry Pi can shut down during nighttime and restart in the morning every day to reduce unnecessary power consumption. The size and capacity of the battery can be also reduced with this function.

6.2 Advanced Image Analysis

Compared to car counters, webcam monitoring is a strategy which has more potential since images can provide much more information. If webcam monitoring systems were implemented at all 15 parking lots, it would be impractical for park rangers to stare at the screen and check the traffic conditions for over 30 webcams. Instead, Artificial Intelligence (AI) could be applied to recognize objects of the captured images. For example, if traffic congestion was detected, AI would automatically send an alert to park rangers so that they could quickly take action.

In addition, another potential of recognizing the images is to recognize the number and types of cars passing through as well as the number of visitors in the images. With this technology to collect data, the static webcam system could also be used as several indicators adopted by National Park Service like Vehicles at one time and People per Viewscape (NPS, 2019a). In conclusion, conducting image recognition using AI could improve the efficiency of the monitoring system and automatically extract valuable information from the images. The Figure shown below is a mock-up of AI implementation.

Figure 34. Mock-up of AI Implementation

6.3 Park Implementation

For the past three years, WPI has investigated the feasibility of webcams and their applicability to address the unique challenges presented by the park's visitor congestion issues. At the conclusion of this project, we believe we have sufficiently demonstrated the feasibility of using static cameras for traffic monitoring to the best of the ability offered by WPI. Dozens of private entities throughout the country already provide technologies similar to those which we determined necessary, many of which would likely be willing and able to customize their pre-existing commercial offerings to match the unique challenges presented by the monitoring components of the Traffic Management Plan.

Figure 35. Process of Project Development

To gauge the interest in and availability of contractors working in this field, we recommend that the park releases a Request for Information, which will help pave the way to receiving proposals and selecting a qualified third party organization to handle the specific details of implementation, deployment, and maintenance of such a complicated system.

Conclusion

Acadia National Park has become one of the most popular national parks in recent years, which leads to congestion in parking lots, key roads, hiking trails and major attractions. Many monitoring strategies in the Final Transportation Plan will be implemented. The 2019 team developed a comprehensive webcam monitoring system including static webcam, mobile webcam and mock-up websites. We proved this system could be an efficient additional monitoring strategies for park rangers and an important information source to visitors helping them better plan their trips to avoid congestion areas.

After three weeks of testing, we verified that our static webcam system can work continuously in the wild while automatically capturing and uploading images of road conditions to the webpage with low maintenance. We also proved the concept of mobile webcam system, which could be installed on the Island Explorer with the capability of taking images at given coordinates autonomously and uploading them online when there is cellular connectivity. A mock-up websites was made to present the current and past conditions of the images of both webcam systems for the use of park rangers and visitors. After evaluating the data and feasibility of both webcam systems, we recommended some potential equipment improvement as well as image recognition algorithms that could improve the efficiency of the monitoring system and automatically extract valuable information. We also suggested that the park could consider sending a Request For Information followed by a Request For Proposal, in order to contract with a third party company for further development.

Overall, our team completed the technical verification phase of a comprehensive webcam monitoring system for Acadia National Park and indicated that it could be a critical component for traffic monitoring and information dissemination. We hope that the National Park Service benefits from this project and proceeds with it. We also look forward to reducing the workload of transportation management for park rangers and improving visitor traffic after implementing webcam monitoring systems.

Reference

Accu-Traffic inc. (2010). Automatic traffic recorder (ATR) counts. Retrieved from http://www.accu-traffic.ca/service/automatic-traffic-recorder-atr-counts

Barrameda, C. E., Vose, T. R., & Rizzo, T. A. (2018). *Congestion management in glacier national park*

Bergquist, Z., Palacios, A. C., & Goklevent, S. (2018). *Cellular connectivity status in acadia national park*

Bruno, M., Cromwick, M., & Feng, Y. (2018). *Feasibility of webcam implementation in acadia national park*

Charles Jacobi. (2016). *Vehicle traffic and visitor use of the*

Cadillac Mountain Summit road 2015. ().

Cosmopulos, E., Gaulin, J., Jauris, H., Morisseau, M., & Quevillon, E. (2017). *Preparing acadia national park for modern tourist congestion*

Dolores Kong, & Dan Ring. (2018). Acadia National Park traffic problems, need for plan, surface at jordan pond. Retrieved from http://acadiaonmymind.bangordailynews.com/2018/05/31/home/acadia-national-park-traffic-problems-need-for-plan-surface-at-jordan-pond/

Downeast Transportation. (2019). Island explorer

. Retrieved from http://www.exploreacadia.com/

Google. (2019). Blue hill overlook. Retrieved from https://www.google.com/maps/place/Blue+Hill+Overlook/@44.3513316,-68.2313737,1130m/data=!3m1!1e3!4m12!1m6!3m5!1s0x4caea3b20ea22925:0x62df220efc555584!2sAcadia+National+Park!8m2!3d44.3385559!4d-68.2733346!3m4!1s0x4caebee5f1d9c393:0xcac7fa7b370d1e05!8m2!3d44.3497047!4d-68.229754

Hallo, J. C., & Manning, R. E. (2009). Transportation and recreation: A case study of visitors driving for pleasure at acadia national park. *Journal of Transport Geography*, (17(6)), 491-499.

Interagency Visitor Use Management Council. (2016). *Visitor use management framework.*

Kong, D., & Ring, D. (2019). Cadillac mountain road closed 54 times in 2018 by crowds in acadia. Retrieved from https://acadiaonmymind.bangordailynews.com/2019/02/03/home/cadillac-mountain-road-closed-54-times-in-2018-by-crowds-in-acadia/

Manning, R., Lawson, S., Newman, P., Hallo, J., & Monz, C. (2014). *Sustainable transportation in the national parks*Ecology & Environmental Studies / Conservation / Transportation.

Monz, C. A., Marion, J. L., Goonan, K. A., Manning, R. E., Wimpey, J., & Carr, C. (2010). Assessment and monitoring of recreation impacts and resource conditions on mountain summits: Examples from the northern forest, USA.*Mountain Research and Development, 30*(4), 332-343. doi:10.1659/MRD-JOURNAL-D-09-00078.1

National Geographic. (2019-03-08T12:17:00-0500). Top 10 most visited national parks. Retrieved from https://www.nationalgeographic.com/travel/national-parks/most-visited-parks-photos/

The National Oceanic and Atmospheric Administration. (2019). Graphical forecasts - caribou, ME. Retrieved from https://graphical.weather.gov/sectors/car.php#tabs

National Park Service. (2018a). Manage congestion - transportation (U.S. national park service). Retrieved from https://www.nps.gov/subjects/transportation/congestion-management.htm

National Park Service. (2018b). *National park service 12/31/2018 land resources division summary of acreage*

National Park System. (2018). Webcams. Retrieved from https://www.nps.gov/acad/learn/photosmultimedia/webcams.htm

National Park Service. (2019a, Mar 26,). Acadia National Park final transportation plan and final environmental impact statement released. *US Official News*

National Park Service. (2019b). Traffic congestion. Retrieved from https://www.nps.gov/acad/traffic.htm

National Park Service. (2019c). Visitation numbers (U.S. national park service). Retrieved from https://www.nps.gov/aboutus/visitation-numbers.htm

National Park System. (2019). *Annual visitation report by years: 2008 to 2018.* (). Retrieved from https://irma.nps.gov/Stats/SSRSReports/National%20Reports/Annual%20Visitation%20By%20Park%20(1979%20-%20Last%20Calendar%20Year)

RSG. (2017). *Cadillac mountain transportation and visitor use model*

timeanddate. (2019). Past weather in Bar Harbor, Maine, USA — july 2018. Retrieved from https://www.timeanddate.com/weather/@4957320/historic?month=7&year=2018

Appendix A: Battery Estimation

		Idle		Transmitting	
Component	**Voltage**	**Current (A)**	**Power (W)**	**Current (A)**	**Power (W)**
Raspberry Pi	5	0.26	1.3	0.26	1.3
Camera	5	0.25	1.25	0.25	1.25
EC25-A	5	0.035	0.175	2.0	10.0
		Total	2.725	**Total**	12.5

Based on an average upload time of 10 seconds, an interval of 120 seconds, and an active period from 4:00am to 8:00pm, we can calculate that the camera is active for 5.5 percent of the day, and idle for the remaining 94.5 percent.

Given these values, we can calculate the average power consumption during the active period with $(2.725 \times 110 + 12.5 \times 10)/120 = 3.47W$ which allows us to determine the number of watt-hours consumed during the day and night.

Window	**Average Power (W)**	**Hours**	**Energy (WH)**
Day	3.47	16	55.52
Night	2.725	8	21.8
		Total	77.32

The SLA1116 battery has a nominal voltage of 12V and a rated capacity of 18AH, giving it a total energy capacity of 216 watt-hours. Due to losses in the power conversion circuitry, we can only maintain an average efficiency of approximately 80 percent. Factoring this in, we have approximately 172.8 watt-hours of usable energy.

$$172.8WH/77.32WH = 2.23\ Days$$

Assuming a total loss of solar power, the camera should be able to function for approximately 2 days, 5 hours, and 30 minutes (53.5 hours). In practice, a number of factors including weather, temperature, partial cloud cover, and electrical interference can greatly alter this number.

Appendix B: Static Camera Bill of Materials

#	Component	Cost	Qty.	Total Cost
1	Weatherproof Box	$33.20	1	$33.20
2	DC-DC Buck Converter	$5.41	1	$5.41
3	Weatherproof 8P8C (Ethernet) Connector *	$15.77	1	$15.77
4	Weatherproof Ethernet Cable *	$18.83	1	$18.83
5	SLA1116 Sealed Lead Acid Battery **	$48.19	1	$48.19
6	Raspberry Pi	$35.00	1	$35.00
7	SMA to u.FL adapter	$4.95	1	$4.95
8	4G/LTE Base Shield V2 for Raspberry Pi	$39.00	1	$39.00
9	Quectel EC25 Mini PCIe 4G/LTE Module	$49.00	1	$49.00
10	50W Solar Panel ***	$52.18	1	$52.18
11	Hydrophobic Coating Spray	$5.39	1	$5.39
12	Solar Charge Controller	$10.99	1	$10.99
13	Standoffs for Raspberry Pi	$1.39	1	$1.39
14	Machine Screws for Raspberry Pi	$0.99	1	$0.99
15	Machine Nuts for Raspberry Pi	$0.99	1	$0.99
16	Female power connector	$4.34	2	$8.68
17	Male power connector	$5.35	2	$5.35
18	LTE Antenna	$8.55	1	$8.55
			Total	$343.86

* *Optional Component*
** *May be substituted for alternative 12V SLA battery*
*** *May be substituted for alternative 12V panel*

Appendix C: Acadia National Park Closures

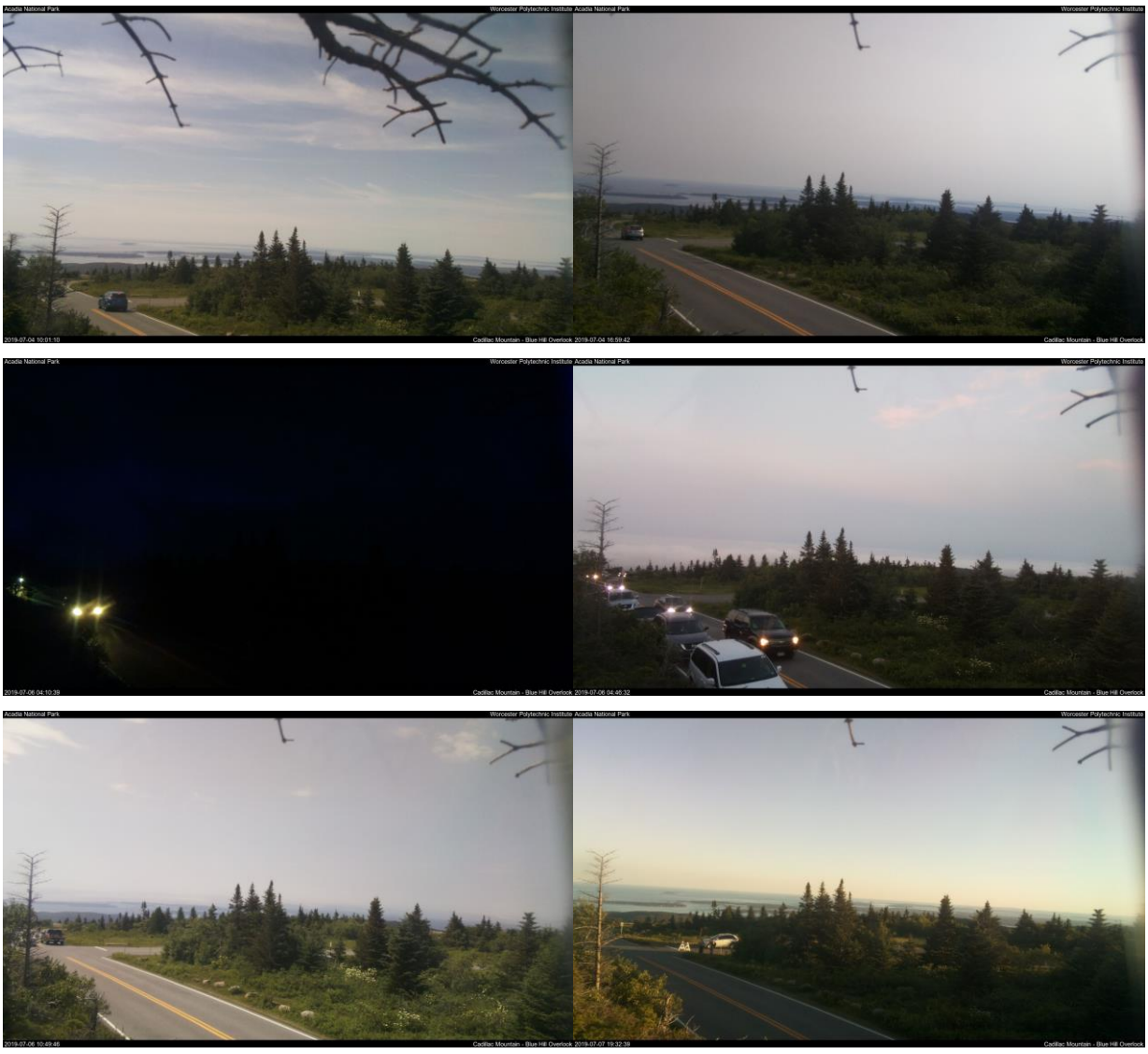
July 1, 2019 - July 22, 2019

Acadia National Park Closures and High Visitation Statistics 2019						
Date	Time of Event	Time Back to Normal Ops	Reason (congestion, MVA, etc.)	Specific Location (Cadillac Summit, Ocean Drive, Schoodic Point, HCVC) *We don't include when parking lots are full	Status Board Update Y/N	Cars turned away
7/1/2019	2001	2020	Congestion	Cadillac mountain road	N	42
7/2/2019	1959	2029	Congestion	Cadillac mountain road	N	97
7/2/2019	1959	2028	Congestion	Bass harbor head lighthouse	N	
7/4/2019	1303	1350	Congestion	Echo Lake Road	N	
7/4/2019	1925	2011	Congestion	BHHL	N	22
7/4/2019	1925	2132	Congestion	Cadillac Summit Rd	Y	
7/5/2019	1730	1828	Congestion	Cadillac Summit Rd	Y	197
7/5/2019	1908	2027	Congestion	Cadillac Summit Rd	y	364
7/5/2019	2005	2045	Congestion	Hwy 233 Entrance	N	34
7/5/2019	2000	2028	Congestion	Bass Harbor Light rd	N	52
7/6/2019	1111	1200	Congestion	Bass Harbor Light rd	N	unknown
7/6/2019	1213	1248	Congestion	Cadillac Mountain road	N	221
7/6/2019	1315	1400	Congestion	Bass Harbor light	N	unknown
7/7/2019	1953	2017	Congestion	Cadillac Summit	Y	66
7/9/2019	1933	2009	Congestion	Bass Harbor light road	N	20
7/13/2019	1952	2015	Congestion	Cadillac Summit	Y	unkown
7/15/2019	1949	2017	Congestion	Cadillac Summit Rd	Y	64
7/16/2019	1144	1215	Congestion	Cadillac Summit Rd	Y	100
7/18/2019	1954	2016	Congestion	Cadillac Summit Rd	N	36
7/20/2019	1958	2011	Congestion	Cadillac Summit Rd	N	15
7/22/2019	1205	1233	Congestion	Cadillac Summit Rd	Y	72
7/22/2019	1236	1244	Congestion	Bass Harbor LH	N	16

Appendix D: Sample Photos of Static Webcam System

Our static webcam took over 9500 images during its working period. It is impossible to attach all the images, so we chose some representative ones. The attached link is the database of all the images we captured: https://nps-cams.wpi.edu/images/

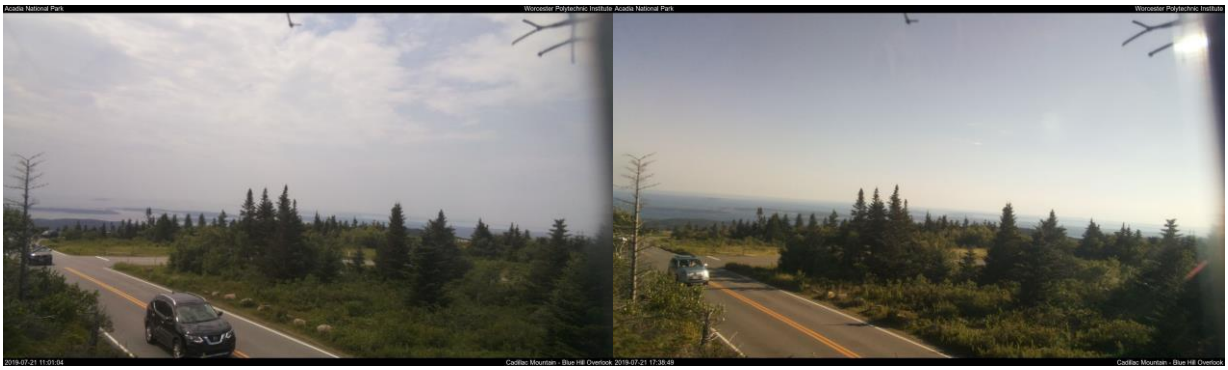
Blue Hill Overlook
July 4, 2019 - July 23, 2019




Acadia National Park
Worcester Polytechnic Institute
2019-07-08 12:45:47
Cadillac Mountain - Blue Hill Overlook
2019-07-09 05:20:41
2019-07-12 07:04:17
2019-07-12 07:08:30
2019-07-12 14:46:01
2019-07-12 18:23:26
2019-07-13 07:25:00
2019-07-13 11:21:29


Acadia National Park
Worcester Polytechnic Institute
2019-07-13 19:56:35
Cadillac Mountain - Blue Hill Overlook
2019-07-14 16:15:21
Cadillac Mountain - Blue Hill Overlook
2019-07-14 16:40:39
2019-07-16 04:55:48
2019-07-17 12:14:23
2019-07-17 16:25:51
2019-07-18 08:49:05
2019-07-18 16:13:31


Acadia National Park
Worcester Polytechnic Institute
2019-07-20 08:02:44
Cadillac Mountain - Blue Hill Overlook
Acadia National Park
Worcester Polytechnic Institute
2019-07-20 19:20:13
Cadillac Mountain - Blue Hill Overlook

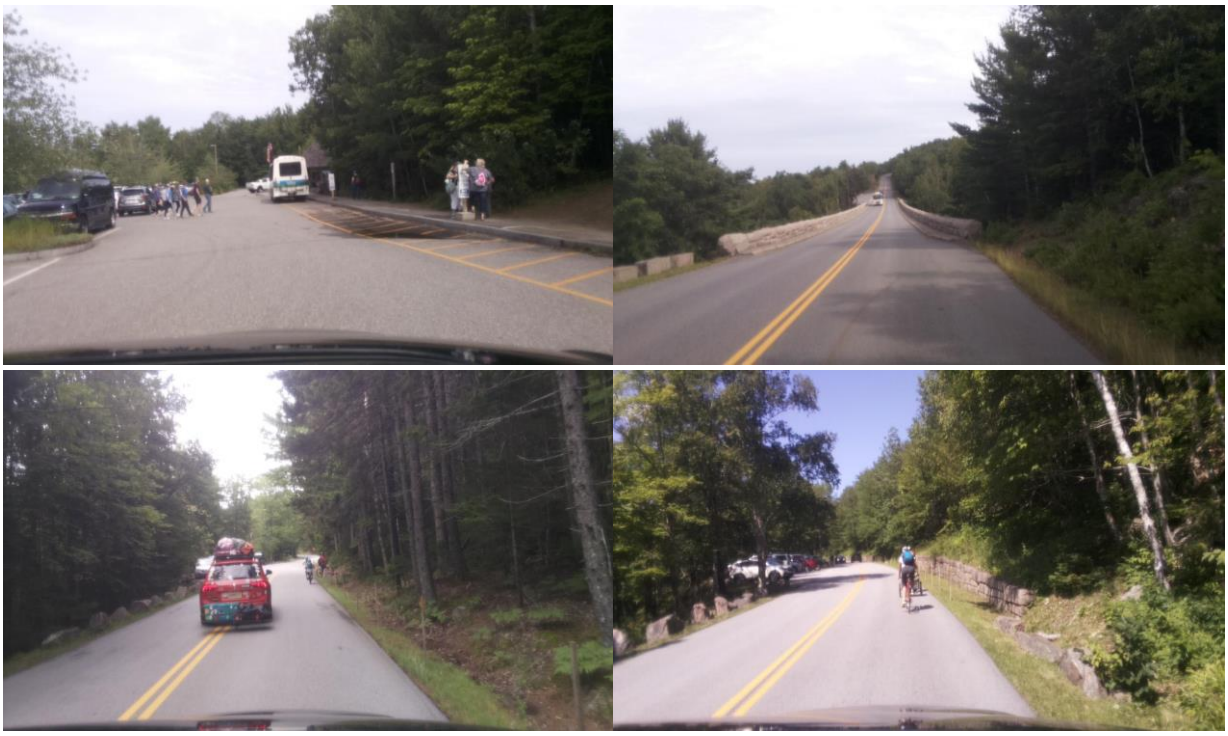

Acadia National Park
Worcester Polytechnic Institute
2019-07-21 11:01:04
Cadillac Mountain - Blue Hill Overlook
Acadia National Park
Worcester Polytechnic Institute
2019-07-21 17:38:49
Cadillac Mountain - Blue Hill Overlook


Acadia National Park
Worcester Polytechnic Institute
2019-07-22 04:57:16
Cadillac Mountain - Blue Hill Overlook
Acadia National Park
Worcester Polytechnic Institute
2019-07-22 17:14:02
Cadillac Mountain - Blue Hill Overlook

Appendix E: Sample Photos of Mobile Webcam System

July 23, 2019 & July 25, 2019


STOP



Appendix F: Source Code

A complete copy of this code along with additional documentation can be found on WPI's GitLab server at https://arcgit.wpi.edu/webcams19

Static Camera Source Code:

camera.py:

```
#!/usr/bin/python3 -u

##
# camera.py - Extreme Webcam Controller 9000
#
# Written by Nicholas Hollander <nhhollander@wpi.edu>
# Written on 2019-05-03
#

from picamera import PiCamera
import datetime
import os
import config
import math
import time
import upload
import subprocess
import ec25

# Print something to separate the logs
datestr = datetime.datetime.now().strftime("%T on %F")
msg = "STARTING UP AT {}".format(datestr)
print("\n  \033[1;33m\u2554{}\u2557".format("\u2550"*len(msg)))
print("  \u2551{}\u2551".format(msg))
print("  \u255A{}\u255D\033[0m\n".format("\u2550"*len(msg)))

#########
# SETUP #
#########

# Print the PID
print("My PID is \033[1;33m{}\033[0m".format(os.getpid()))
print("Kill with \033[1m~/scripts/kill_children.sh {}\033[0m".format(os.getpid()))
# Change the working directory
print("\033[1mChanging working directory...\033[0m")
os.chdir("/home/pi")
print("Working directory is \033[1m{}\033[0m".format(os.getcwd()))
print("\033[1mConnecting to EC25...\033[0m")
ec25.init()

# Read the configuration
print("\033[1mLoading Configuration...\033[0m")
if not config.read():
        print("\033[31mNo configuration loaded - terminating\033[0m")
        exit()

# Initialize EC25
print("\033[1mConnecting to EC25\033[0m")
ec25.init(config.config['ec25']['enable_gps'])

# Set up and connect to the camera
print("\033[1mSetting up camera...\033[0m")
camera = PiCamera()
camera.resolution = (
```

```
        config.config['camera']['resolution']['x'],
        config.config['camera']['resolution']['y'])
print("Using resolution to \033[1m{}\033[0m".format(camera.resolution))

# Get the interval
print("\033[1mConfiguring interval...\033[0m")
start_time = config.config['interval']['start_time']
stop_time  = config.config['interval']['stop_time']
start_hour = math.floor(start_time / 60)
start_min  = start_time % 60
stop_hour  = math.floor(stop_time / 60)
stop_min   = stop_time % 60
interval   = config.config['interval']['interval']
print("Taking pictures every \033[1;33m{}\033[0m minutes, starting at \033[1;33m{:02d}:{:02d}\033[0m and
ending at \033[1;33m{:02d}:{:02d}\033[0m!".format(interval, start_hour, start_min, stop_hour, stop_min))

#############
# MAIN LOOP #
#############

while True:
        # Get the current time
        ctime = datetime.datetime.now()
        # Extract components of the time
        year =   ctime.year
        month =  ctime.month
        day =   ctime.day
        hour =   ctime.hour
        minute = ctime.minute
        second = ctime.second
        ttime =  (hour * 60) + minute
        # Compare time to start of day and end of day bounds
        if ttime < start_time:
        # It's too early for this!
        time.sleep((start_time - ttime) * 60)
        continue
        elif ttime > stop_time:
        # It's too late for this!
        print("\033[35mIt's too late to be taking pictures!\033[0m")
        if config.config['power']['disconnect_at_night']:
        print("Disabling EC25 (Goodnight)")
        ec25.set_functionality(0)
        # Wait until midnight
        time.sleep(((24 * 60) - ttime) * 60)
        continue
        # Enable EC25 only if it is not already enabled
        if not ec25.functionality == 1:
        print("Enabling EC25")
        ec25.set_functionality(1)
        # Generate the timestamped filename
        fname = "data/{}-{:04d}-{:02d}-{:02d}_{:02d}-{:02d}-
{:02d}.jpg".format(config.config['capture']['name_base'],year,month,day,hour,minute,second)
        print("Saving image to \033[1m{}\033[0m".format(fname))
        camera.capture(fname)
        # Compress and annotate the image
        subprocess.run(['/home/pi/scripts/annotate.sh',fname,'{:04d}-{:02d}-{:02d}
{:02d}:{:02d}:{:02d}'.format(year,month,day,hour,minute,second),config.config['capture']['name_full']])
        # Upload the image and delete it (on successful upload)
        upload.upload(fname, True)
        print("Waiting \033[1m{}\033[0m seconds before taking next picture".format(interval * 60))
        time.sleep(interval * 60)
```

config.json:

```
{
```

```
    "ec25": {
        "enable_gps": true
    },
    "camera": {
        "resolution": {
            "x": 3280,
            "y": 2464
        }
    },
    "power": {
        "disconnect_after_upload": true,
        "disconnect_at_night": true
    },
    "interval": {
        "start_time": 240,
        "stop_time": 1200,
        "interval": 2
    },
    "server": {
        "username": "static-camera",
        "key_file": "/home/pi/.ssh/id_npscams",
        "hostname": "nps-cams.wpi.edu",
        "directory": "/datastorage/images/",
        "retry_count": 8,
        "retry_delay": 5
    },
    "capture": {
        "name_full": "Cadillac Mountain - Blue Hill Overlook",
        "name_base": "mountain",
        "quality": 80
    }
}
```

config.py:

```
##
# config.py - Configuration tools 'n stuff
#
# Written by Nicholas Hollander <nhhollander@wpi.edu>
# Written on 2019-05-03
#

import json
import os
import errno

# Configuration dictionary
config = dict()

##
# Read the configuration.
# This function reads the configuration file from disk and saves the variables
# to the configuration map.
def read():
    global config
    try:
        # Open the file
        cfile = open('config.json','r')
        # Parse the file
        config = json.load(cfile)
        # Close the file
        cfile.close()
        return True
    except IOError as e:
        # Something went wrong while reading the file
        print("\033[1;31mERROR:\033[0m "
```

```
        "Something went wrong while reading \033[1mconfig.json\033[0m: {}({}):{}"
        .format(errno.errorcode[e.errno], e.errno, os.strerror(e.errno)))
        return False
```

ec25.py:

```
##
# ec25.py - Helper functions for the Quectel EC25-A module
#
# Written by Nicholas Hollander <nhhollander@wpi.edu>
# Written on 2019-05-16
#

import serial
import time
import threading
from ast import literal_eval
import traceback

# Global connection instance
cmd_connection = None
# GPS feedback connection
gps_connection = None

# Run flag - set to False to stop threads
run = True

######################
## STATUS VARIABLES ##
######################

# Sim card insertion status
sim_card_inserted = None
# Sim card auto-update enabled
sim_card_inserted_auto = None
# Scanned carriers
available_carriers = None
# GPS data
gps_data = None
# Functionality
functionality = 1

# Handlers
handlers = dict()

##############################################
## MAGICAL SETUP AND CONFIGURATION METHODS ##
##############################################

# Connect
def init(enable_gps=False):
    global cmd_connection
    global gps_connection

    # Connect
    print("Connecting to ec25 control interface")
    cmd_connection = serial.Serial("/dev/ttyUSB2", timeout=1)

    # Configure the error mode (verbose errors)
    cmd_connection.write("AT+CMEE=2\r\n".encode())

    # Start the input thread
    primary_thread = threading.Thread(target=__primary_thread,name='PrimaryThread')
    primary_thread.start()

    # Check for GPS
```

```
    if enable_gps:
        print("Connecting to gps feedback interface")
        # Enable GPS
        do_enable_gps()
        # Open the GPS connection
        gps_connection = serial.Serial("/dev/ttyUSB1", timeout=4)
        # Start the gps input thread
        gps_thread = threading.Thread(target=__gps_thread,name='GPSThread')
        gps_thread.start()

    # Done
    print("Connection complete")

# Enable GPS
def do_enable_gps():
    # Disable GPS first (leftover session bug)
    cmd_connection.write("AT+QGPSEND\r\n".encode())
    time.sleep(1)
    # Send the gps enable command
    cmd_connection.write("AT+QGPS=1\r\n".encode())

#########################
## THREADS AND PARSING ##
#########################

# Primary query response thread
def __primary_thread():
    while run:
        # Read input - The timeout specified when the serial port was opened
        # means that this method will return a blank message after one second
        # of waiting if no new messages are available
        line = cmd_connection.readline().decode()
        if line == "":
            continue
        # Ignore non-update messages, which always begin with a plus.
        if not line[0] == "+":
            continue
        # TODO: Parse command here

# GPS data thread
def __gps_thread():

    # GPS coordinate decode method
    def gps_decode(coord):
        x = coord.split(".")
        head = x[0]
        tail = x[1]
        deg = head[0:-2]
        minute = head[-2:]
        return int(deg) + (float(minute) * (1/60))

    global gps_data
    while run:
        # Get the next line of GPS data
        line = gps_connection.readline().decode()
        print(line)
        # TODO: Add parsing for other gps data types, incl. altitude
        # GPRMC -> Recommended minimum specific GPS/Transit data
        if line.startswith("$GPRMC"):
            # Split the data
            sdata = line.split(",")
            if sdata[2] == "V":
                # Satellite data is not available
                gps_data = None
                continue
            # Parse the data
            time = sdata[1][0:2] + ":" + sdata[1][2:4] + ":" + sdata[1][4:6]
            lat = gps_decode(sdata[3])  #latitude
```

```
            dirLat = sdata[4]           #latitude direction N/S
            lon = gps_decode(sdata[5])  #longitute
            dirLon = sdata[6]           #longitude direction E/W
            speed = sdata[7]            #Speed in knots
            trCourse = sdata[8]         #True course
            date = sdata[9][0:2] + "/" + sdata[9][2:4] + "/" + sdata[9][4:6] # date
            variation = sdata[10]       #variation
            dc = sdata[11].split("*")   # degree checksum
            degree = dc[0]              #degree
            checksum = dc[1]            #checksum
            # Combine to form the gps data
            gps_data = (time,lat,dirLat,lon,dirLon,speed,trCourse,date,variation,degree,checksum)

############################
## FUNCTIONS AND QUERIES ##
############################

##
# Sets the device functionality level
# 0 - Minimum Functionality
# 1 - Full Functionality
# 4 - Disable RX and TX
def set_functionality(level):
    global functionality
    # Verify
    if not (level == 0 or level == 1 or level == 4):
        print("Invalid functionality setting [{}]".format(level))
    else:
        cmd_connection.write("AT+CFUN={}\r\n".format(level).encode())
        # Hacky fix - change this to use something nicer later.  Or never.
        functionality = level
```

start.sh:

```
#!/bin/bash

# Start and daemonize the script
/home/pi/code/camera.py 2>&1 >> /home/pi/log/log.txt &
```

upload.py

```
##
# upload.py - Tool for uploading images to the web server
#
# Written by Nicholas Hollander <nhhollander@wpi.edu>
# Written on 2019-05-16

import pysftp
import threading
import config
import traceback
import time
import os
import ec25

# List of threads.
threads = []

##
# Upload a file.
# This function attempts to upload a file to the server, retrying on failure
# until the retry limit is reached.  If the file is uploaded, it is deleted
# from the device.  If the upload fails completely, then the file will be
# left on the device.
```

```
def upload(fname, delete=False):
    # Start the upload thread
    t = threading.Thread(target=__upload, args=(fname,delete))
    threads.append(t)
    t.start()

##
# Upload Thread.
# Performs the nitty gritty of the above function without blocking program
# execution.
def __upload(fname, delete):
    # Load configuration information
    username = config.config['server']['username']
    keyfile = config.config['server']['key_file']
    hostname = config.config['server']['hostname']
    directory = config.config['server']['directory']
    retry_count = config.config['server']['retry_count']
    retry_delay = config.config['server']['retry_delay']
    powersave_enabled = config.config['power']['disconnect_after_upload']
    # Upload loop
    for i in range(0, retry_count):
        # Print some information
        print("Uploading file \033[1m{}\033[0m to \033[1m{}@{}:{}\033[0m (Attempt \033[33m{}\033[0m of
\033[33m{}\033[0m)".format(fname, username, hostname, directory, i + 1, retry_count))
        # Catch connection errors
        try:
            # Connect and upload the file
            connection = pysftp.Connection(hostname, username=username, private_key=keyfile)
            connection.chdir(directory)
            connection.put(fname, preserve_mtime=True)
            # Delete file from local device
            if delete:
                os.remove(fname)
            print("Upload of \033[1m{}\033[0m complete!".format(fname))
            if powersave_enabled:
                print("Disabling EC25 (Upload complete)")
                ec25.set_functionality(0)
            return
        except Exception as e:
            # Print the stack trace
            print("\033[1;31mError:\033[0m Something went wrong while uploading \033[1m{}\033[0m:
(\033[1;31m{}\033[0m)".format(fname, type(e).__name__))
            print("\033[1;31m=== Begin traceback ===\033[0m")
            traceback.print_tb(e.__traceback__)
            print("\033[1;31m=== End traceback ===\033[0m")
            # Delay
            delay_time = retry_delay * (i + 1)
            print("Retrying upload in \033[1m{}\033[0m seconds.".format(delay_time))
            time.sleep(delay_time)
```

annotate.sh

```
#!/bin/bash

# Check for arguments
if [ -z "$1" ] || [ -z "$2" ] || [ -z "$3" ] || [ -z "$4" ]; then
    printf "\033[1;31mError: Missing arguments\033[0m\n"
    echo "Usage:"
    echo "$0 <filename> <timestamp>"
    exit
fi

# Annotates the heck out of images
convert "$1" \
        -resize 1920x1080^ -gravity center -extent 1920x1080 \
```

```
        -bordercolor black -border 0x25 -fill white -pointsize 20 -font Arial \
        -gravity southeast -annotate +5+0 "$3" \
        -gravity northwest -annotate +5+0 "$4" \
        -gravity northeast -annotate +5+0 'Worcester Polytechnic Institute' \
        -gravity southwest -annotate +5+0 "$2" \
        -quality 80 "$1"
```

Server and Website Code:

get_latest.py:

```
#!/usr/bin/python3

import json
import re
import glob
import os

try:
        # Determine the image stream to select the image from
        stream = os.environ.get("QUERY_STRING", "invalid-stream")
        # Remove invalid characters
        stream = re.sub('[^a-z]','',stream)
        # Truncate the name
        stream = stream[:40]
        # Get a list of files
        files = glob.glob('images/{}-*.jpg'.format(stream))
        # Find the newest file
        latest_file = max(files, key=os.path.getctime)

        # Deliver the response
        print('')
        print(latest_file)
except:
        # Error!
        print('')
        print('error.jpg')
```

webcams.js:

```
/*!
 *  NPS Cameras Automatic Update Script
 *
 *  Written by Nicholas Hollander <nhhollander@wpi.edu>
 */

function register_updater(element) {
    // Get the stream name
    var sname = element.getAttribute("wc-stream");
    // Get the update interval
    var interval = parseInt(element.getAttribute("wc-interval"));
    // Create the update function
    var update = function() {
        console.log("Updating [" + sname + "]...");
        // Get the update from the server
        var request = new XMLHttpRequest();
        request.onreadystatechange = function() {
            if(this.readyState == 4 && this.status == 200) {
```

```
                // Update the image
                console.log("Got response for [" + sname + "]!");
                element.src = request.responseText;
            }
        };
        request.open("GET", "get_latest.py?" + sname);
        request.send();
    };
    // Create the update function
    setInterval(update, interval);
    // Initial invocation
    update();
}

// Find elements that can be updated
console.log("Finding images");
elements = document.querySelectorAll("img[wc-stream][wc-interval]");
elements.forEach(function(element) {
    // Register the magical update gizmo
    register_updater(element);
    // Register the click handler
    element.addEventListener("click", image_clicked);
});

// Handle an image being clicked
function image_clicked(event) {
    // Get the close frame
    var cframe = document.getElementById("cframe");
    // Check if enlarged
    if(event.target.className.includes("enlarged")) {
        console.log("Collapsing frame");
        // Remove the enlarged attribute
        event.target.className = event.target.className.replace("enlarged","");
        // Hide the frame
        cframe.className = cframe.className.replace("enlarged","");
    } else {
        console.log("Expanding frame");
        // Update the class name
        event.target.className += " enlarged";
        // Show the frame
        cframe.className += " enlarged";
    }
}
```

Mobile Camera Code:

The base of this code was provided courtesy of the 2019 Glacier National Park Webcam team. Some debug, support, and configuration scripts have been omitted from this submission.

autorun.sh

```
#!/bin/bash
# Test is the process is running using the if statement.
#su pii

# Kill existing python processes
killall python3

while true;
do
 if ! ps -ef | egrep '[g]pscamdaemon.sh' ;
    then
    echo "program gpscamdaemon is not running - run the program"
```

```
        /home/pi/gpscamdaemon.sh > /home/pi/gpscamdout.txt &
else
        echo "program gpscam is running - nothing to do"
fi

if ! ps -ef | egrep '[f]tpdaemon.sh'  ;
then
        echo "program ftpdaemon is not running - run the program"
        /home/pi/ftpdaemon.sh > /home/pi/ftpdout.txt &
else
        echo "program ftpdaemon is running - nothing to do"
fi

sleep 5
done
```

gps-init.py

```
#!/usr/bin/python3

import serial
import subprocess
import time

print("Connecting to serial control device")
serial = serial.Serial("/dev/ttyUSB2")

print("Configuring GPRS")
# Due to a bug in the EC25, it is possible for ghost sessions to be left behind
# when the device restarts or encounters an error.  These ghosts will prevent
# GPS readings from being taken and new GPS sessions from being created until
# they are killed.
serial.write("AT+QGPSEND\r\n".encode()) # Close any existing sessions
time.sleep(2)
serial.write("AT+QGPS=1\r\n".encode()) # Enable GPS
serial.close()

print("Starting GPS daemon")
subprocess.call(['sudo','killall','gpsd'])
subprocess.call(['sudo','gpsd','/dev/ttyUSB1','-F','/var/run/gpsd.soc'])
#subprocess.call(['cgps','-s'])

print("Done")
```

main.py

```
from gps3 import gps3
from picamera import PiCamera
from time import sleep
import time
#from playsound import playsound
import csv
import math
from datetime import datetime

# Calculate the distance between two points on the surface of the earth
def haversine(c1, c2):
        R = 6372800 # Radius of earth in meters
        lat1, lon1 = c1
        lat2, lon2 = c2
        phi1, phi2 = math.radians(lat1), math.radians(lat2)
```

```
        dphi = math.radians(lat2 - lat1)
        dlambda = math.radians(lon2 - lon1)
        a = math.sin(dphi/2)**2 + math.cos(phi1)*math.cos(phi2)*math.sin(dlambda/2)**2
        return 2*R*math.atan2(math.sqrt(a), math.sqrt(1 - a))

def takepic(name,camera):
        camera.start_preview()
        sleep(.25)
        tstamp = datetime.now().strftime("%Y-%m-%d_%H-%M-%S")
        camera.capture('/home/pi/Desktop/webcam/pics/'+name+"."+tstamp+".jpg")
        camera.stop_preview()
        print('pic taken: '+name)
        return
def inrange(truelat,truelon,data_stream):
        c1 = (truelat,truelon)
        c2 = (float(data_stream.TPV['lat']),float(data_stream.TPV['lon']))
        dist = haversine(c1, c2)
        print("Distance is \033[33m{}m\033[0m".format(dist))
        return dist < 10

#       return ((abs(float(data_stream.TPV['lon'])-
truelon)<=.0001)and(abs(float(data_stream.TPV['lat'])-truelat) <= .0001))
def main(useftp = False ):
        # Parse the data points file
        pointfile = open('/home/pi/points.csv','r')
        csv_reader = csv.reader(pointfile)
        points = []
        for row in csv_reader:
        gpoint = {
        'lat': float(row[0]),
        'lon': float(row[1]),
        'name': row[2],
        'radius': float(row[3])
        }
        print("Added new point at \033[33m{}\033[0m, \033[33m{}\033[0m named \033[33m{}\033[0m with an
active radius of \033[33m{}m\033[0m".format(gpoint['lat'], gpoint['lon'], gpoint['name'],
gpoint['radius']))
        points.append(gpoint)
        pointfile.close()
        camera = PiCamera()
        gps_socket = gps3.GPSDSocket()
        data_stream = gps3.DataStream()
        gps_socket.connect()
        gps_socket.watch()
        # Create the debug data output file for storing distance information
        fname = datetime.now().strftime("/home/pi/debug_%Y-%m-%d_%H-%M-%S.csv")
        debug_data_file = open(fname,'w')
        for new_data in gps_socket:
        if new_data:
        data_stream.unpack(new_data)
        if(data_stream.TPV['lon'] == 'n/a'):
                print('searching for gps signal...')
                continue
        print('Longitude= ',data_stream.TPV['lon'],' +/- ', data_stream.TPV['epx'], ' meters')
        print('Latitude = ', data_stream.TPV['lat'], ' +/- ', data_stream.TPV['epy'], 'meters')
        print('Time = ', data_stream.TPV['time'])
        print('GPS Speed = ', data_stream.TPV['speed'], ' +/-', data_stream.TPV['eps'], ' m/s')
        print('================================')
        current_lat = float(data_stream.TPV['lat'])
        current_lon = float(data_stream.TPV['lon'])
        dist_data = []
        for point in points:
                c1 = (point['lat'],point['lon'])
                c2 = (current_lat,current_lon)
                dist = haversine(c1,c2)
                dist_data.append(str(dist))
                print("Point \033[33m{}\033[0m, dist \033[33m{:.2f}m\033[0m".format(point['name'],
dist))
```

```
                if dist < point['radius']:
                print("Point matches!")
                takepic(point['name'], camera)
                break
        debug_data_file.write(','.join(dist_data))
        debug_data_file.write("\n")
        debug_data_file.flush()


print('===GLACIER NATIONAL PARK MOBILE WEBCAM SYSTEM===')
main()
```

ftp.py

```
import pysftp
import subprocess
from time import sleep
from pathlib import PosixPath
import os

########################
#####FTP LOGIN INFO#####
########################
USERNAME = 'REDACTED'
PASSWORD = 'REDACTED'

def getSignalStrength():
        return 999
        python3_command = "python2 /home/pi/Desktop/webcam/sigstr.py"
        process = subprocess.Popen(python3_command.split(), stdout=subprocess.PIPE)
        output, error = process.communicate()  # receive output from the python2 script
        print(output.strip().decode('ascii'))
        strength = int(output.strip().decode('ascii')) #hacking
        return strength

photodir = PosixPath('/home/pi/Desktop/webcam/pics/')
#print(list(photodir.glob('*.jpg')))
#print(str(list(photodir.glob('*.jpg'))[0]))
while(True):
        print("main loop: checking for images")
        while list(photodir.glob('*.jpg')): #is there something in the list?
        print("something's in the list.")
        snapshot = list(photodir.glob('*.jpg'))
        print(snapshot)
        #main program loop: check for new images
        if (getSignalStrength() > -70):
        sftp = pysftp.Connection('nps-cams.wpi.edu', username=USERNAME, password=PASSWORD)
        print("connected")
        sftp.chdir('/datastorage/images/glacier/')
        print("changed directory")
        sftp.put(str(snapshot[0]))  # upload first file in list to public/ on remote
        print("put file successfully")
        os.remove(str(snapshot[0])) # remove file from system
        print("removed file from local pc")
        #playsound(happy)
        sftp.close()
        else:
        print("signal strength not strong enough...")
        #playsound(sad)


        sleep(10)
```